$2.95 USA

Australia	A $ 4.75	New Zealand	NZ $ 6.50
Singapore	S $ 9.45	Hong Kong	H $23.50
Malaysia	M $ 9.45	Sweden	30:-SEK

'68' MICRO JOURNAL

VOLUME VI ISSUE II • Devoted to the 68XX User • February 1984

"Small Computers Doing Big Things"

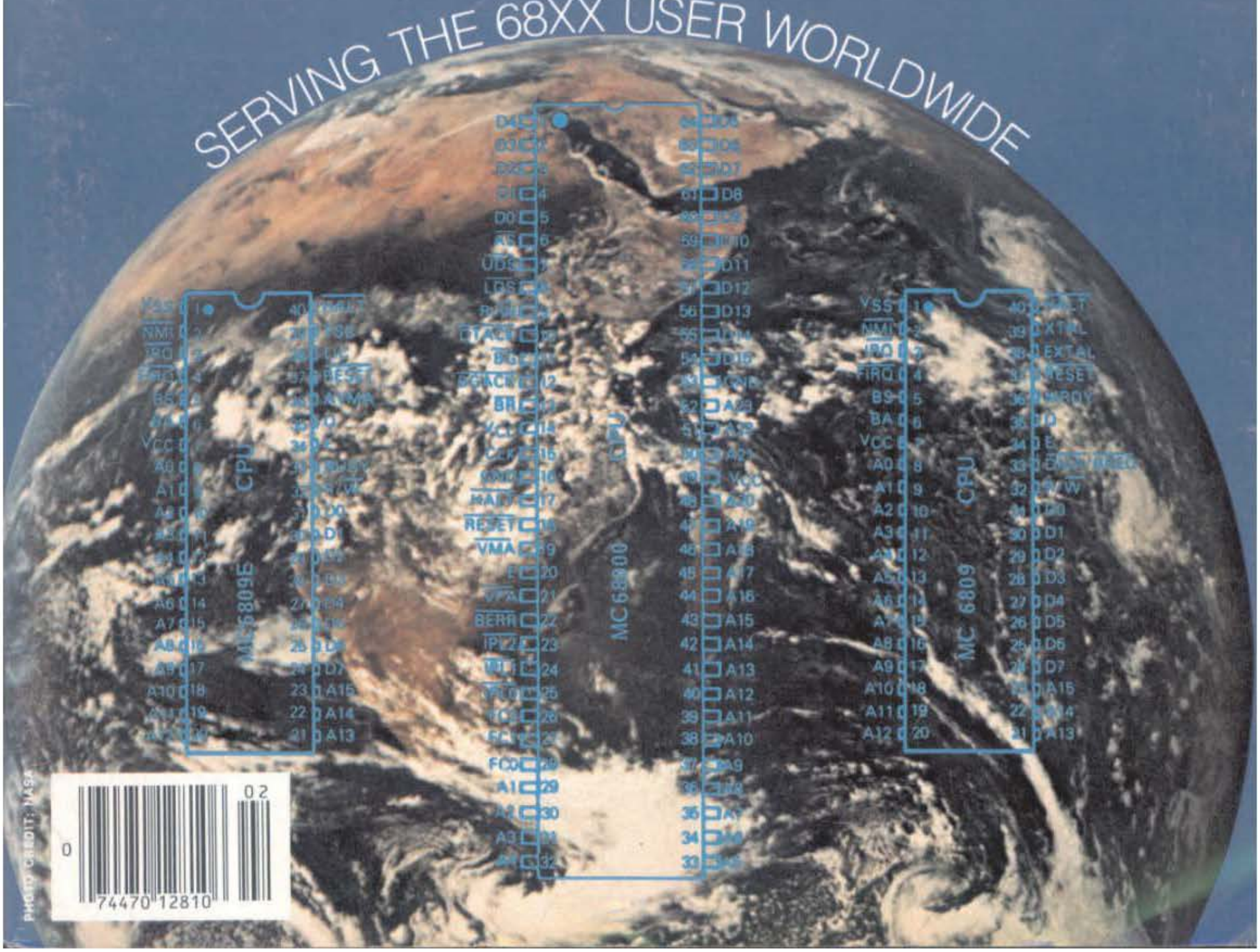
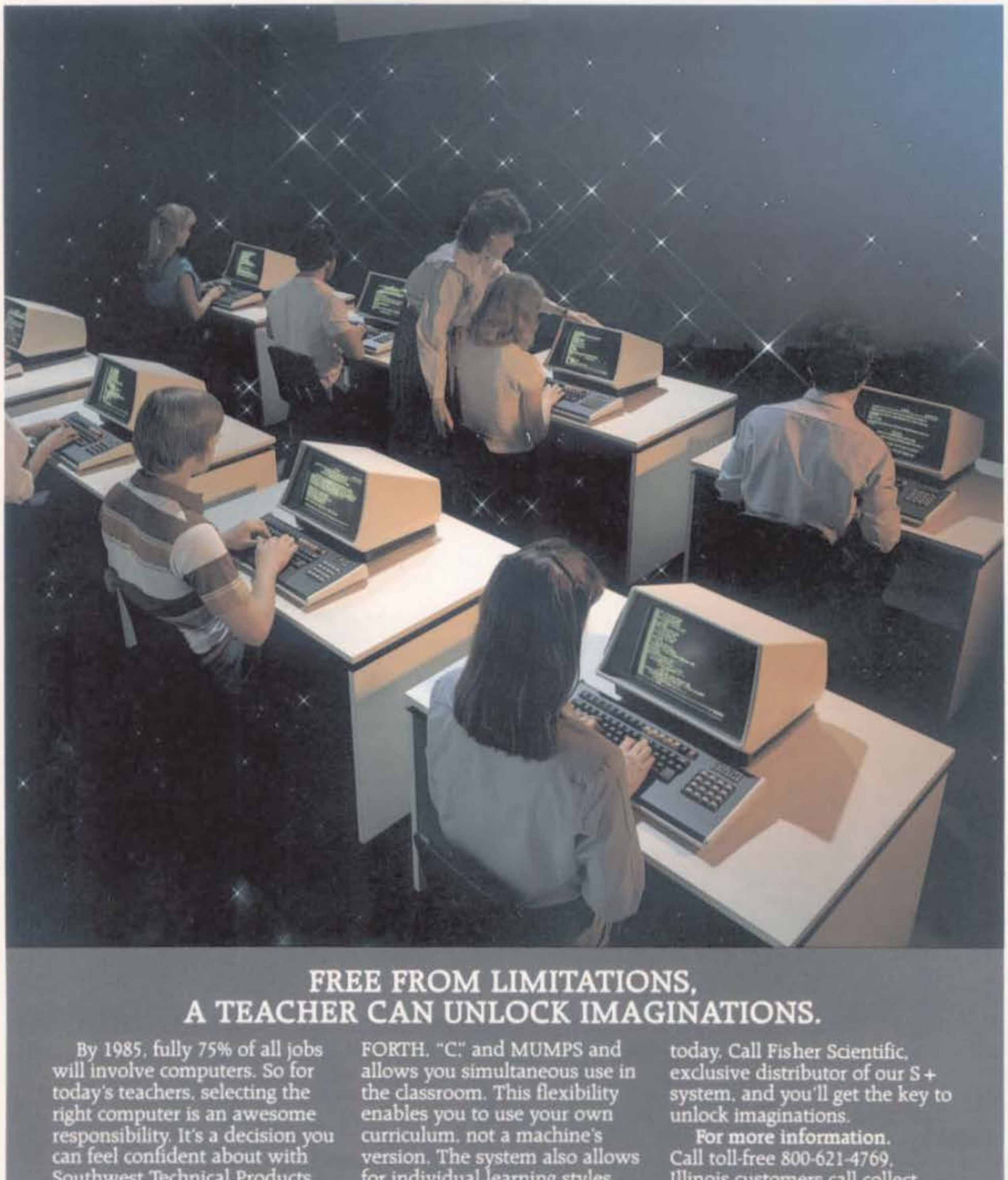


PHOTO CREDIT: NASA



FREE FROM LIMITATIONS, A TEACHER CAN UNLOCK IMAGINATIONS.

By 1985, fully 75% of all jobs will involve computers. So for today's teachers, selecting the right computer is an awesome responsibility. It's a decision you can feel confident about with Southwest Technical Products Corporation's S+ computer system. The S+ is specially designed for a student's tomorrow and a teacher's today. It's built tough. It's simple to operate. You can teach without becoming a hardware specialist.

The S+ is fluent in a variety of languages: BASIC, PASCAL, COBOL, FORTRAN, PILOT, FORTH, "C," and MUMPS and allows you simultaneous use in the classroom. This flexibility enables you to use your own curriculum, not a machine's version. The system also allows for individual learning styles. Students go as fast and as far as the teacher chooses.

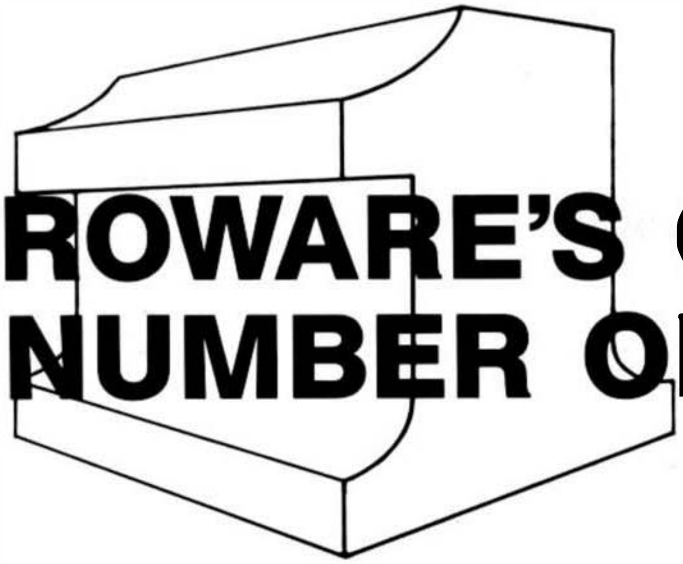
Even though it's priced like a smaller personal computer, the S+ has the multi-task power and expansion abilities of a professional computer. It can grow with you and your students.

Tomorrow is in your hands, today. Call Fisher Scientific, exclusive distributor of our S+ system, and you'll get the key to unlock imaginations.

For more information. Call toll-free 800-621-4769. Illinois customers call collect (312) 378-7770.

Fisher Scientific
An ALLIED Company

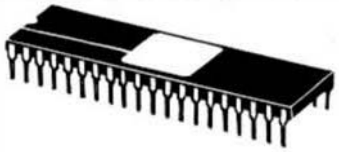
Educational Materials Division
4901 W. LeMoyne St.,
Chicago, Illinois 60651



MICROWARE'S OS-9 IS NUMBER ONE.

OS-9 NOW HAS THE LARGEST USER COMMUNITY

More users now run OS-9 on their 6809 computers than all other operating systems combined. This outstanding success story was no accident — it's due to OS-9's technical excellence backed up by outstanding Microware support. OS-9's Unix-type architecture and totally modular design gives your computer more power and versatility. OS-9 also gives you more possibilities for customization so you can tailor your system exactly to your needs. And aren't flexibility and performance the reasons you chose a 6809 computer to begin with?



OS-9 HAS BEEN CHOSEN BY OVER 50 6809 SYSTEM MANUFACTURERS

OS-9 is now offered as a standard operating system by almost every 6809 system manufacturer, and has been designed into an amazing variety of dedicated systems and products including personal and business computers, process control systems, data and telecommunications systems, and more. In all, over 50 companies and organizations have obtained OS-9 distribution licenses including such well-known names such as General Motors, NASA, Fujitsu, Western Electric, Motorola, Sykes Datatronics, Eastman Kodak, Thomson-CSF, and Tandy Corp.

OS-9 GIVES YOU A SOFTWARE BASE TO BUILD ON

Whatever your application, OS-9 speaks your language! Microware offers BASIC09, an Extended/ Structured Basic, a complete C Compiler, a full ISO Pascal Compiler, the ANSI Standard CIS Cobol Compiler, plus Relocatable Macro Assembler. These high performance programming languages are all fully implemented and deliver unmatched performance and outstanding features. Additionally, OS-9 compatible applications packages such as word processors, screen editors, spreadsheets, business software, and utilities are offered by a rapidly growing number of third-party software houses.

PLUS OUTSTANDING MICROWARE SUPPORT: WE KEEP IN TOUCH WITH YOU

Even when you have the best software and documentation, there can be times when you need questions answered. That's why Microware is committed to giving OS-9 users the best possible **personalized** service. Here are some of the ways we deliver solid support:

- A Software Support Hotline for direct access to our technical staff
- "Pipelines", our free quarterly newsletter
- OS-9 User Seminars, the annual OS-9 community gathering
- a liberal update policy for new releases

Microware does business on a person-to-person basis. When you call you'll find yourself speaking with someone who's both knowledgeable and genuinely interested in helping.

YOU CAN COUNT ON OS-9 NOW AND IN THE FUTURE

Microware is not standing still — we're firmly committed to continuing support for the 6809 and we will continue to introduce exciting new software products for the 6809. We will soon announce OS-9/68000 and programming languages for the 68000 which will be upward compatible with 6809 versions, so if and when you are ready for the 68000 your OS-9 software can go with you.

MICROWARE®

Microware Systems Corporation
Box 4865 • Des Moines, IA 50304
515-279-8844 Telex 910-520-2535

OS-9 and BASIC09 are trademarks of Microware and Motorola. Unix is a trademark of Bell Labs.

OS-9: BETTER BY DESIGN

'68' MICRO JOURNAL

Portions of the text for 68 MICRO JOURNAL was prepared using the following furnished hard/software.

COMPUTERS-HARDWARE

Southwest Technical Products
219 W. Rhapsody
San Antonio, Texas 78216
S09-5/8 DMF disk-CDS1-8212W-Sprint 3 Printer
-
GIMIX Inc.
1337 West 37th Place
Chicago, IL 60609
Super Mainframe-OS9-FLEX-Assorted Hardware

EDITORS-WORD PROCESSORS

Technical Systems Consultants, Inc.
111 Providence Road
Chapel Hill, NC 27514
FLEX-Editor-Processor
-
Great Plains Computer Company, Inc.
PO Box 916
Idaho Falls, ID 83401
STYLO-Mail Merge

Editorial Staff

Don Williams Sr.	Publisher
Larry E. Williams	Executive Editor
Tom E. Williams	Production Editor
Robert (Bob) Nay	Color Editor

Administrative Staff

Mary Robertson	Office Manager
Joyce Williams	Accounting
Carolyn Williams	Subscriptions
Penny Williams	File Management

Contributing Editors

Ron Anderson
Norm Commo
Peter Dibble
Dr. Theo Elbert
William E. Fisher
Dr. E.M. Pass

Special Technical Projects

Clay Abrams K6AEP
Tom Hunt

CONTENTS

Vol.VI, Issue II FEB'84

Send All Correspondence To:

Computer Publishing Center
68 MICRO JOURNAL
5900 Cassandra Smith
PO Box 849
Hixson, TN 37343
615 842-4600

68' Micro Journal is published 12 times a year by Computer Publishing Inc. Second Class Postage Paid ISSN 0194-5025 at Hixson, Tenn. and additional entries. Postmaster: send Form 3579 to 68' Micro Journal, PO Box 849, Hixson, Tennessee.

SUBSCRIPTION RATES
USA
1-Year $24.50 2-Years $42.50 3-Years $64.50
FOREIGN
See Page 52

Items Submitted for Publication

Articles submitted for publication should be accompanied by the authors full name, address, date and telephone number. It is preferred that articles be submitted on either 5 or 8 inch diskette in TSC Editor format or STYLO format. All diskettes will be returned.

The following TSC Text Processor commands ONLY should be used (due to our proportional processor): .sp space, .pp paragraph, .fi fill and .nf no fill. Also please do not format within the text with multiple spaces. The rest we will enter at time of editing.

STYLO commands are all acceptable except the .pg page command, we print edited text files in continous text.

All articles submitted on diskettes should be in TSC FLEX™ format, either FLEX2 6800, or FLEX9 6809 any version.

If articles are submitted on paper they should be on white 8X11 bond or better grade paper. No hand written articles (hand written or drawn art accepted). All paper submitted articles will be photo reproduced. This requires that they be typed or produced with a dark ribbon (no blue), single spaced and type font no smaller than 'elite' or 12 pitch. Typed text should be approximately 7 inches wide (will be reduced to column width of 3 1/2 inches). **Please use a dark ribbon!**

All letters to the editor should also comply with the above and bear a signature. Letters of 'gripes' as well as 'praise' are solicited. We attempt to publish all letters to the editor verbatim, however, we reserve the right to reject any submission for lack of 'good taste'. We reserve the right to define what constitutes 'good taste'.

Advertising: Commercial advertisers please contact the 68 Micro Journal advertising department for current rate sheet and requirements.

Classified: All classified must be non-commercial. Maximum 20 words per classified ad. Those consisting of more than 20 words should be figured at .35 cents per word. 20 words or less $7.50 minimum, one time, paid in advance. No classified ads accepted by telephone.

TO REALIZE THE FULL POWER & PERFORMANCE OF THE 6809, LOOK TO GIMIX.

GIMIX OFFERS YOU A VARIETY OF SS50 BUS COMPONENTS AND SYSTEMS.

OS-9 GMX III

The GMX 6809 CPU III and OS-9 GMX III. A Multi-user, Multi-tasking package for the ultimate in System Performance plus protection of the system and other users from crashes caused by errors in individual users programs

#01 (CPU & Software) NEW!	$1698.01

INTELLIGENT I/O PROCESSOR BOARDS increase system throughput by reducing interrupts to the host, buffering data transfers, and data preprocessing. Prices include on board firmware. Requires system drivers.

#11 3 port RS232 Serial (SS30) NEW!	$498.11
#12 4 port Parallel (SS50)	$538.12
OS-9 GMXIII drivers . . (included when purchased with GMX III package)	$200.00

OS-9 Level 2 users -- contact GIMIX for system requirements and availability.

192K GMX III #79 SYSTEMS: All include GMX 6809 CPU III and OS-9 GMX III (#01); a #11 3 port intelligent serial I/O & cables; #19 Classy Chassis; 192KB Static RAM, #68 DMA controller, all necessary cables, power regulators, and filler plates. The OS-9 Editor, Assembler, Debugger, BASIC-09, and RUNB are included.

#79 with dual 40 track DSDD drives	$5998.79
#79 with dual 80 track DSDD drives NEW!	$6298.79
#79 with #88 8" Dual Drive Disk System	$7598.79
#79 with #90 19MB Winchester subsystem & one 80 track DSDD drive	$8998.79

UniFLEX for the GMX 6809 CPU III and Intelligent I/O boards is in development.

OS-9 GMX I; OS-9 GMX II; FLEX; and UniFLEX

The #05 GIMIX 6809 PLUS CPU board			$578.05
Options: GMX DAT	$35.00	SWTPC DAT	$15.00
9511A	$312.00	9512	$265.00

#49 64KB GHOST SYSTEM includes: #05 CPU; #19 Classy Chassis; 64 KB static RAM; a #43 2 port serial card & cables; #68 DMA Controller; all necessary cables, power regulators, and filler plates; GMXBUG monitor; FLEX; and OS-9 GMX I. You can software select either FLEX or OS-9. The OS-9 Editor, Assembler, Debugger, BASIC-09, and RUNB are also included.

#49 with dual 40 track DSDD drives	$4398.49
#49 with dual 80 track DSDD drives	$4698.49
#49 with #88 8" Dual Drive Disk System	$5998.49
#49 with #90 19MB Winchester subsystems & one 80 track DSDD drive	$7398.49

#39 128KB SYSTEM includes: #05 CPUwDAT; #19 Classy Chassis; 128KB of static RAM; a #43 2 port serial card & cables; #68 DMA Controller; all necessary cables, power regulators, and filler plates; GMXBUG monitor; FLEX; and OS-9 GMX II. You can software select either FLEX or OS-9. The OS-9 Editor, Assembler, Debugger, BASIC-09, and RUNB, and GMX-VOISK for FLEX are included.

#39 with dual 40 track DSDD drives	$4998.39
#39 with dual 80 track DSDD drives	$5298.39
#39 with #88 8" Dual Drive Disk System	$6598.39
#39 with #90 19MB Winchester subsystem & one 80 track DSDD drive	$7998.39

UniFLEX- available at extra cost, requires 8" or Winchester drives. A signed license agreement with TSC is required before shipment.

You can add to any GIMIX system RAM, I/Os and other options, or substitute non-volatile RAM. GIMIX will customize to your needs.

COMING SOON: Contact GIMIX for price and availability on 40MB and 72MB Winchester (5¼") drives, removeable pack Winchesters, 256KB static RAM boards.

All GIMIX systems are guaranteed for 2MHz operation. GIMIX systems include documentation for all boards and software in a GIMIX binder. **ALL DRIVES ARE 100% TESTED AND ALIGNED BY GIMIX.**

ALL BOARDS AND SYSTEMS ARE ASSEMBLED, BURNED-IN, AND TESTED. GOLD-PLATED BUS CONNECTORS ARE USED.

TO ORDER BY MAIL: SEND CHECK OR MONEY ORDER OR USE YOUR VISA OR MASTER CHARGE. Please allow 3 weeks for personal checks to clear. U.S. orders add $5 handling if order is under $200.00. Foreign orders add $10 handling if order is under $200.00. Foreign orders over $200.00 will be shipped via Emery Air Freight COLLECT, and we will charge no handling. All orders must be prepaid in U.S. funds. Please note that foreign checks have been taking about 8 weeks for collection so we would advise wiring money, or checks drawn on a bank account in the U.S. Our bank is the Continental Illinois National Bank of Chicago, 231 S. LaSalle Street, Chicago, IL 60693, account #73-32033. Visa or Master Charge also accepted.

EXPORT MODELS: ADD $30 FOR 50Hz. POWER SUPPLIES.

GIMIX Inc. reserves the right to change pricing, terms, and product specifications at any time without further notice.

ALL PRICES ARE F.O.B. CHICAGO

Choose from GIMIX' wide variety of system components.

The **GIMIX CLASSY CHASSIS #19** consists of a heavyweight aluminum cabinet, constant voltage ferro-resonant power supply, and SS50 Mother board with baud rate generator board . . . $1398.19

Triple Disk regulator card and cables	$88.22	Baud rate generator card	$88.93
Missing cycle detector	$38.23	Filler plates	$14.92
Back panel connector plates (specify)	$8.60	50 Hz. option	$30.00

MEMORIES (GIMIX uses only Static RAM)

#67 64KB NMOS STATIC RAM board	$478.67
#64 64KB CMOS STATIC RAM board w/battery back-up	$568.64
#34 8K PROM board	$98.34
#32 16 socket PROM/ROM/RAM board	$238.32

I/O Boards (see above for Intelligent I/Os)

#41 Single port serial, RS232/20ma. current loop	$88.41
#43 2 port serial, RS232	$128.43
#46 8 port serial, RS232	$318.46
#42 2 port parallel	$88.42
#45 8 port parallel	$198.45
#50 serial, RS232, RS422, RS423	$244.50
#52 SSDA serial, RS232, RS422, RS423	$254.52
#54 ADLC serial, RS232, RS422, RS423	$288.54
Each cable with connectors for back panel mounting (specify board)	$24.95

DISK CONTROLLERS

#68 DMA (featured in all systems above)	$588.68
#28 dbl. dens. programmed I/O (5" drives only)	$298.28
#58 single dens. programmed I/O (5" and/or 8" drives)	$226.58
#48 same as #58 but for 5" drives only	$198.48
Cable sets: 8" with Back Panel connector	$29.25
for two 8" external drives	$44.26
for two 5" drives	$34.96

SOFTWARE: GIMIX exclusive versions of OS-9/GMX I, II, III & FLEX are for GIMIX hardware only. All versions of OS-9 require the #68 controller.

When ordered with any controller, FLEX is . . . $30.00

GMXBUG PROMs and manual			$98.65
Boot or Video boot PROM	$30.00	UNIFLEX boot PROM	$90.00
OS-9 GMX I	$200.00	OS-9 GMXII	$500.00
Editor	$125.00	Assembler	$125.00
BASIC-09	$200.00	RUNB	$100.00

DISK DRIVES FOR GIMIX SYSTEMS – complete with cables and power regulators.

5" DSDD 40 track			2 for $900.00
5" DSDD 80 track			2 for $1300.00
#88" Dual 8" DSDD drives, cabinet, power supply, & cables			$2698.88
Cabinet only	$848.18	220V 50Hz. Option, add	$30.00
Filler plate	$14.83	Cable for 2 drives	$44.82
Cable for 4 drives	$67.84	Cable for cabinet to mainframe	$45.81

WINCHESTER SUBSYSTEMS: for use only in GIMIX systems with #68 DMA controller.

#90: includes one 19MB drive, interface, and Software	$3588.90
#91: includes two 19MB drives, interface and Software	$5288.91

Contact GIMIX for price and availability of other forthcoming subsystems.

OTHER BOARDS

#76 GHOST 80X24 VIDEO BOARD			$398.76
#66 50 pin Protoboards	$56.66	#33 30 pin Protoboards	$38.33
#03 6800 CPU			$224.03
#06 6800 CPU with timers	$288.06	Baud rate option, add	$30.00
#08 RELAY DRIVER (board, bracket, transformer, and 31 relays)			$1128.08
#86 - #08 (board, bracket, transformer, without relays)			$538.86
#85 OPTO board			$348.85
WINDRUSH EPROM PROGRAMMER			$375.00
3" Binder	12.00	2" Binder	$9.00

GIMIX DOES NOT GUARANTEE PERFORMANCE OF ANY GIMIX SYSTEMS, BOARDS OR SOFTWARE WHEN USED WITH OTHER MANUFACTURERS PRODUCT.

DON'T SEE IT??? ASK! OUR BROCHURE HAS MORE COMPLETE DISCRIPTIONS AND SPECS. PHONE OR WRITE TODAY FOR YOUR COPY.

BASIC-09 and OS-9 are trademarks of Microware Systems Corp. and MOTOROLA, Inc. FLEX and UniFLEX are trademarks of Technical Systems Consultants, Inc. GIMIX, GHOST, GMX, CLASSY CHASSIS, are trademarks of GIMIX, Inc.

GIMIX inc.

1337 WEST 37th PLACE • CHICAGO, ILLINOIS 60609
(312) 927-5510 • TWX 910-221-4055

Microware presents 4 new OS-9 software packages.

1 LEVEL II PRINT SPOOLING SYSTEM

This versatile package gives your OS-9 Level Two System a complete print spooling management capability for time-sharing applications. Features of the spooling system are:

- *Handles up to seven independent spooling devices and queues with "print on first available device" feature.*
- *Prints large block header pages between listings with date, time, user name and job name.*
- *Multiple listing copy option.*
- *Complete forms change capability for each job and device.*
- *Prints formatted or unformatted listings.*
- *Status command displays print queues and status.*
- *User can kill or change priority of queued jobs.*

*Available **only** for OS-9 Level Two Systems.*

Suggested List Price: $150.00 Manual Only: $15.00

2 RMA RELOCATABLE MACRO ASSEMBLER

At last — a full feature relocatable macro assembler and linkage editor for OS-9. RMA permits sections of assembly language programs to be independently assembled to "relocatable object files". The linkage editor takes any number of program sections and/or library sections and combines them into a single executable OS-9 memory module. Global data (including indexed and direct addressing modes) and program references are automatically resolved in the process. The macro facility permits commonly used statement sequences to be defined, then used within the program with appropriate parameter substitution. RMA also supports conditional assembly and library source files.

Suggested List Price: $200.00 Manual Only: $20.00

3 OS-9 FILE HANDLER TOOLBOX

Introducing a special toolbox for OS-9 users who do a lot of file manipulation! A collection of 12 useful OS-9 command programs; Most can be used as "filters" using OS-9 pipeline facilities. Included are:

***D** — unformatted directory listing with "wild card" matching*
***Compress** — does character compression on text files.*
***Expand** — restores a "compressed" file to the original state.*
***Split** — breaks a file into smaller files.*
***Space** — indents lines with optional spacing between lines.*
***Code** — decodes any key on a keyboard to hex.*
***Qsort** — quick sort for small files, directories, etc.*
***Pr** — versatile formatted file printing utility.*
***Tr** — transliterates text pattern to substitution pattern.*
***Grep** — searches file for a pattern and prints matching lines.*
***Xmode** — same "tmode" except changes are made to the device descriptor.*
***Count** — counts words, lines, or characters within a text file.*

Suggested List Price $85.00

4 ENTERTAINMENT PACK I

A collection of games and other interesting programs that are not only entertaining but serve as good instructional examples of Basic09 programming techniques. All programs include complete Basic09 source files and can be easily edited to run on standard alphanumeric or graphics terminals.

***Blkjak** — A Vegas-rules blackjack game.*
***Clk** — graphical display of a wall clock on your terminal.*
***Dogs** — Greyhound racing with simulated graphics.*
***Eliza** — Basic09 version of the famous artificial intelligence simulation of natural language dialogue with a psychiatrist.*
***Haiku** — Program that creates original "haiku" prose.*
***Quest** — a mini-"Adventure" game.*
***Rats** — find your way out of a computer-generated maze — from a rat's point of view.*
***Towers** — a graphical display of the solution to the "Tower of Hanoi" puzzle.*

Suggested List Price: $85.00

MICROWARE.

Microware Systems Corporation
P.O. Box 4865 • Des Moines, IA 50304
515-279-8844 • Telex 910-520-2535

OS-9 and Basic09 are trademarks of Microware and Motorola.
Unix is a trademark of Bell Labs.

"More help than any other thing
I have ever read, except 68 Micro
Journal, of course. Come on with
the C book on Norm Commo's column.
D

FLEX™ USER NOTES
THE 6800-6809 BOOK

By: Ronald W. Anderson
As published in 68 MICRO JOURNAL™

SWELL! - Best buy EVER in a
6800/6809 hook.
JTW

The publishers of 68 MICRO JOURNAL are proud to announce the publication of Ron Anderson's **FLEX USER NOTES**, in book form. This popular monthly column has been a regular feature in 68 MICRO JOURNAL SINCE 1979. It has earned the respect of thousands of 68 MICRO JOURNAL readers over the years. In fact, Ron's column has been described as the 'Bible' for 68XX users, by some of the world's leading microprocessor professionals. Now all his columns are being published, in whole, as the most needed and popular 68XX book available. Over the years Ron's column has been one of the most popular in 68 MICRO JOURNAL. And of course 68 MICRO JOURNAL is the most popular 68XX magazine published.

As a SPECIAL BONUS all the source listing in the book will be available on disk for the low price of: FLEX™ formatonly — 5" $12.95 — 8" $16.95 plus $2.50 shipping and handling. if ordered with the book. If ordered separately the price of the disks will be: 5" $17.95 — 8" $19.95 plus $2.50 shipping and handling.

Listed below are a few of the **TEXT** files included in the book and on diskette.

All TEXT files in the book are on the disks.

LOGO.C1	File load program to offset memory — ASM PIC
MEMOVE.C1	Memory move program — ASM PIC
DUMP.C1	Printer dump program — uses LOGO — ASM PIC
SUBTEST.C1	Simulation of 6800 code to 6809, show differences — ASM
TERMEM.C2	Modem input to disk (or other port input to disk) — ASM
M.C2	Output a file to modem (or another port) — ASM
PRINT.C3	Parallel (enhanced) printer driver — ASM
MODEM.C2	TTL output to CRT and modem (or other port) — ASM
SCIPKG.C1	Scientific math routines — PASCAL
U.C4	Mini-monitor, disk resident, many useful functions — ASM
PRINT.C4	Parallel printer driver, without PFLAG — ASM
SET.C5	Set printer modes — ASM
SETBAS1.C5	Set printer modes — A-BASIC
	(And many more)

**Over 30 TEXT files included in ASM (assembler) — PASCAL — PIC (position independent code) TSC BASIC-C, etc.

NOTE: .C1,.C2, etc. = Chapter 1, Chapter 2, etc.

This will be a limited run and we cannot guarantee that supplies will last long. Order now for early delivery.

Finally I don't have to type in all
that source code, a GREAT, GREAT idea.
Thanks and good luck, you guys are the
best!
TOA

Foreign Orders Add $4.50 S/H

Softcover — Large Format

Book only: **$7.95** + $2.50 S/H

With disk: 5" **$20.90** + $2.50 S/H

With disk: 8" **$22.90** + $2.50 S/H

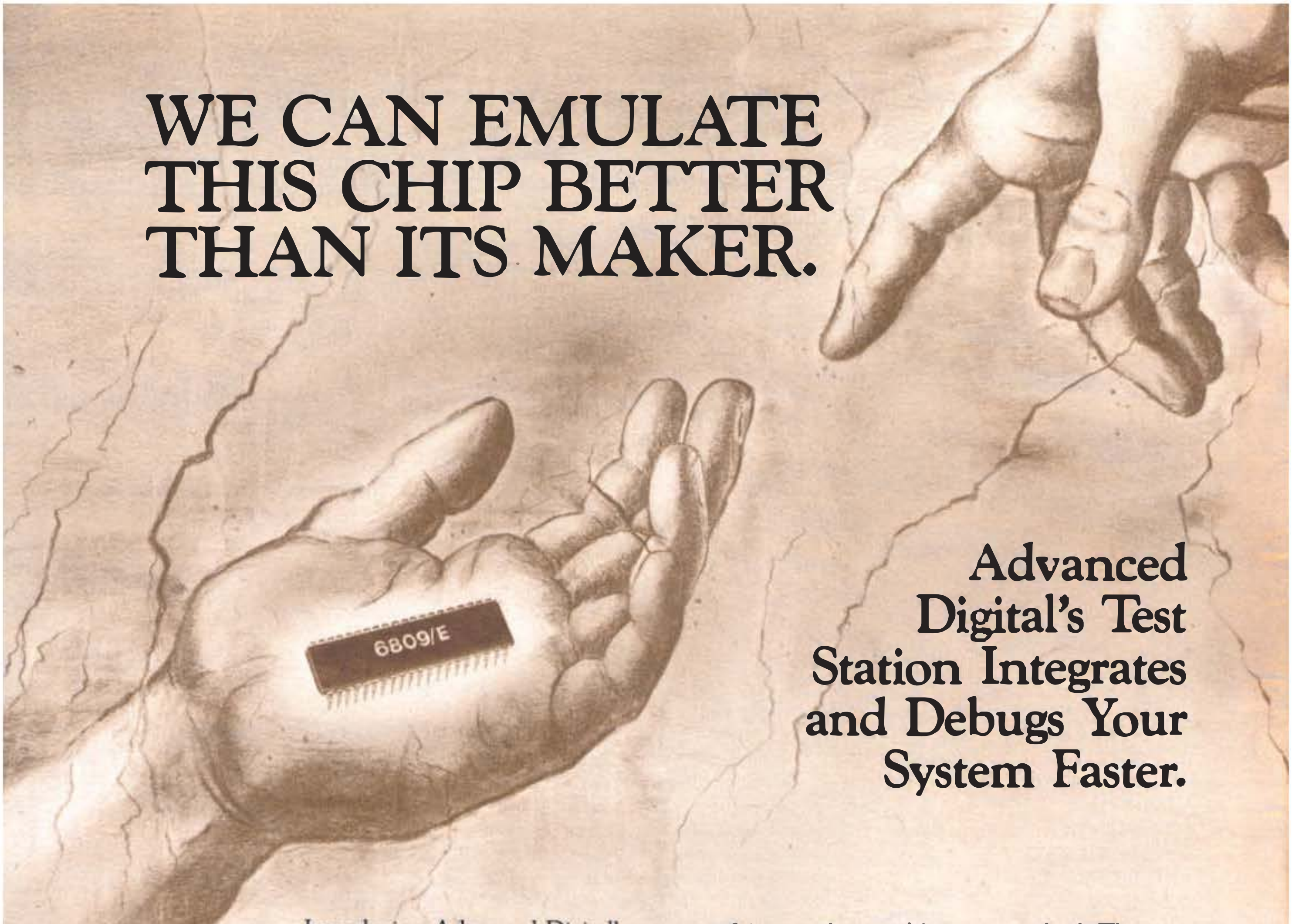
I will keep this (FLEX USER NOTES) book
always beside my computer. I have not
ever, with any magazine, received as much
information as this one book, and 68 MJ.
DIB

See your local S50 dealer/bookstore or order direct from:

Computer Publishing Inc.
5900 Cassandra Smith Rd.
Hixson, TN 37343
(615) 842-4601

VISA

™FLEX is a trademark of Technical Systems Consultants


6809/E

TMP (Total Management Planning) OS-9 SOFTWARE

THESE OFF-THE-SHELF APPLICATION PACKAGES HAVE CHANGED THE PICTURE FOR PROFESSIONAL SOFTWARE USERS

Here are the comprehensively supported OS-9 application software packages that discerning professionals are selecting today. Each one runs stand-alone or in combination with other TMP packages.

TMP/CALC is a *new-generation* "Electronic Spreadsheet" package that goes a quantum leap beyond ordinary Calc software. Superior speed, and extra features like Dynamic Calculations and Dynamic Overlay, mean you can change a cell *anywhere* on the spreadsheet and get automatic updates wherever they belong. No more "wrap-around" since output to the printer is formatted. "Help" screens prompt without erasing data, and the rows and columns on any worksheet are limited *only* by available memory.
ONLY $250

TMP/Manager is a structured database manager, and is the ultimate driving force behind any total *Management/Marketing Planning* system. Its built-in select/sort/merge module — and optimized machine code — makes it super-fast, and provides unusually nimble data manipulation characteristics. Other outstanding features are detailed prompting, easy identification of data fields, and an excellent security system.
ONLY $500

TMP FreeForm fills the void between a structured DBMS and word-processing: it's an "electronic index card" package with an endless list of valuable applications such as parts cataloging, look-up inventories and listings of *any* kind. Type anything on up to 32,000 electronic "cards" of up to 9 pages each that reside in "file drawers" (databases). Up to 13 key-words can call each page.
ONLY $150

TMP/Front-End integrates any or all of the packages described above, and integrates TMP with word-processors, BASIC and other high-level languages.
NO CHARGE with TMP/Manager

TMP T.M.

TMP PACKAGES ARE EXCEPTIONALLY FRIENDLY. EACH COMES WITH MULTI-MEDIA TRAINING AND OUTSTANDING DOCUMENTATION

AUDIO CASSETTE TRAINING TAPES are provided with each TMP software package. These crisp, professionally produced training aids take a new user through "hands-on" exercises that instantly build confidence, and minimize reference to manuals.

CLEAR DOCUMENTATION is specially prepared for each TMP package. Remarkably readable because they are conversational in style and logically organized, this series of manuals *further* assures quick user comfort and productivity with TMP software. Includes sections on how to best utilize the system; integrating user-written software with TMP packages, and other bonus topics.

VIDEOTAPE PROGRAMS are available to TMP Dealers. They dramatically take the viewer into real business environments for a close look at how a TMP software package is utilized. Video programs can be personalized for specific organizations. Programs may be specified for most popular tape formats.

THE BOTTOM-LINE IS THIS: Software is only as effective as its training and documentation. TMP packages are *incomparable* in both areas.

RUN TMP IN A SMOKE SIGNAL CHIEFTAIN™ COMPUTER TO EXPERIENCE THE KIND OF SPEED AND POWER YOU CAN EASILY AFFORD *TODAY*

Endurance-Certified Chieftains are consistently among the fastest computers in CPU and I/O speeds according to the widely respected BENCHMARK REPORTS. That, combined with almost legendary reliability, makes Chieftains the logical choice in computers using OS-9 operating systems.

Configurations range from floppy-disk systems to multi-user, multi-tasking Winchester hard disk systems with tape back-up; performance approaching mainframes at prices you can live with.

TMP software packages for most OS-9 systems and Chieftain computers are offered exclusively from Smoke Signal. Inquiries from non-Smoke Signal dealers are invited. Call (213) 889-9340.

SMOKE SIGNAL®
Chieftain Computers
31336 Via Colinas • Westlake Village, CA 91362
(213) 889-9340

Flex User Notes

Ronald W. Anderson
3540 Sturbridge Court
Ann Arbor, MI 48105

NOTE

Please note that this column is being printed out of order. One of three things happened. I never mailed this one, it was lost in the mail, or the staff at '68' Micro somehow lost it. The benchmark portion of this was written as a more considered response to the article by David Shearer in the June '83 issue, and was done after I found a copy of the Byte article that listed the original timing results for their Benchmark program to find prime numbers. I wrote a letter to the authors of the original and to David Shearer, but neither has responded.

What Happened to the 68000?

I'll start out by saying that the title is a question that I will be asking, and not answering. I note with some chagrin that Cromemco has a couple of new systems out. One of them contains a 68000 and runs a variant of UNIX as the operating system. I've seen nothing or less in the pages of '68' in the way of SS-50 bus and 68000 getting together. Helix seems to be the only manufacturer that is supporting a 68000 system on the "extended" S-50 bus, which they call the S-64 bus.

No one is advertizing software for a 68000 in these pages, perhaps because there is no bus, standard or otherwise, around which to supply software. Are the 68XX equipment suppliers going to let Cromemco and others put the 68000 on the S-100 bus before we have anything going on the S-50? What's wrong guys? Is the apparent market too small to bother with? Did all the "biggies" jump in ahead of you? Perhaps the 68000 is viewed as fitting into a business computer system to replace or compete with the MINI's, and as being too costly to interest any hobbyists.

A friend of mine has bought the Motorola 68000 evaluation board, and we have been waiting for a couple years now, for a cross assembler that doesn't cost more than the evaluation board. "Hand" coding of machine language was easy on the 6800, a good bit harder on the 6809, and considerably more a chore on the 68000, though my friend has done some hand coding and verified that the system does run.

(Late note: LLoyd I/O, supplier of Crasmb, the cross assembler series, has a test version of a 68000 cross assembler that runs under Flex, now out to some selected testers for evaluation. It should be available soon.)

I received some timing information for a test program run on a 68000 system a year or more ago, that was rather disappointing. It was run in the only available Pascal at that time, and the program ran faster in a 6809 Pascal version on a 2 MHZ system. The 68000 was running at 4 MHZ at that time.

While it was inevitable that the 6809 would eventually do away with the 6800 for the most part, I don't see that happening with the 68000. There will be applications around for a good long time for which the 6809 is very well suited in terms of capability and speed, and it will be a much more economically feasable solution for a number of applications for a long time to come. On the other hand, there are applications for which the 68000 is very well suited. When are we hobbyists (and industrial users) going to have an alternative to the "monster bus" that Motorola has created and the very high priced but very capable development system that supports it?

I'd like to quote a recent letter from a reader here. "The 68000 is a fantastic machine. Do you know where it sits on the SS-50/64 bus though? The only machine advertised so far is the HELIX, and no mention has been made about software. I'm not sure what HELIX owners are supposed to do. What are Gimmix, TSC, and Microware doing about the 68000 (and when)?" How about it you suppliers? Another letter indicates that the writer simply assumes that no software will be available for the 68000. "...I'm planning to develop that on a 68000 machine, and I don't quite have the hardware finished yet, let alone an OS...". (Operating System, that is). Since I haven't cleared these quotes with either of the writers, and I'm not sure they want their names published, they will be anonymous for now.

Compilers

I know, I know, you are groaning "not again!". This month though, since I've started in an editorial direction, I thought I'd continue in that direction. My thinking has changed a bit in the past couple years regarding compilers and what they should be. I wrote a letter to one supplier asking why on earth he would spend his time patching BASIC and generating a completely non-standard version of it for his compiler system, when he could be designing a compiler for a standard language. That supplier must have given up on me on the spot, since he has never written to me again. The product was good (I have it), though only available in the 6800 version for FLEX. At the time I thought the approach was bad. This was back in the days when hobbyists were all in the BASIC camp, and perhaps that was the reason for the choice.

Since then, I have seen FORTH, Pascal, and "C" versions by the dozen(s). FORTH has been well adapted to run on the microcomputers. Pascal, if left "standard" has no ability to drive hardware. All the implementations have added features that allow variables at absolute addresses, or some variation of PEEK and POKE to allow data transfer to an I/O port. "C" allows using pointers to addresses for data transfer, and it is rather self sufficient to run on a micro. It is generally customized for a particular hardware system via library files written to tie it to that system.

It seems to me that the library approach is a much better way to customize the language for a particular system. The language remains standard and the library is the link to each individual computer system on which it runs. In all fairness, and in retrospect, I have to admit that STRUBAL was set up that way, with a very standard compiler section and a set of drivers to adapt it to various operating systems.

Since I wrote that letter about the BASIC compiler a couple years ago, I've reconsidered the facts, and have reached a different conclusion. The standard languages were pretty much designed to run on large computer systems that have batch processing. (That is, I think, the reason for Pascal's lack of string handling capabilities). Most microcomputer systems are much more user interactive than large mainframe systems. Since they are different, perhaps the programming languages ought to be different also. Microcomputer hardware is used a great deal in applications where there is no terminal, no disk drive, and no printer. Frequently in industrial applications the only output might be control signals to solid state relays to run a machine, and perhaps some status lights or indicators of some sort. Surely Pascal is not the language of choice for such application programs.

Perhaps some of the recent non-standard languages that have been "tailored" to small computer systems programming or to control systems programming, do really make more sense for the microcomputer. If I am writing a program to run a coffee vendor, do I really care that I can't send it to someone with a different microprocessor to run? Obviously, that is an advantage for me. It will be harder for someone else to steal my program! What I really want is efficiency of compile time and memory usage for the final object code. If the price for that is being non standard, and perhaps having to think a bit more so the compiler can be simpler and smaller, that is perfectly fine.

Obviously, if I am writing the world's best screen oriented editor, I want to write it so that it can be adapted to run on all the other systems based on other processors. In that case I want a standard language, and I will be most careful to use only standard features and not extensions of the language. My conclusion must be that there is a place for both approaches. The best approach will vary with the situation.

MORE TIME TRIALS

Last month I commented on David Shearer's article in the June issue comparing the 6809, FLEX9, and the available software with the Z-80, CP/M and its available software. I reported on the benchmark execution times for Windrush (James McCosh's) "C" compiler (10 seconds) and Windrush PL/9 (14 seconds). (I've since reduced the PL/9 time to 12.5 seconds). I later coded the algorithm in GSPL (my review should have appeared by now) from Workman Associates. It executed in 18 seconds (all on a 2 MHZ system). All three of these look very good compared to the top CP/M compilers run on a 4 MHZ Z-80.

After reporting those findings, I received a note from Lucidata indicating that they had run the benchmark with vastly different results. Just in case the report from Lucidata doesn't (or didn't) get printed, they report that the benchmark will run in their latest version on a 2 MHZ system in 260 seconds contrary to the reported time of 735 seconds. The version of Lucidata Pascal that I have is not quite the current one, but the Pascal program listed here runs in 250 seconds on a 2 MHZ system, which leads me to believe that the time reported by Nigel Bennee of Lucidata, is very accurate, (and that I have the correct algorithm).

Since I had gotten started on this project, I decided to try OmegaSoft Pascal and see how it would fare in these trials. The current version of OMS came in at a very respectable 14 seconds. Just for kicks I tried their old version 1 and it ran the same program in 55 seconds. I guess there was quite an improvement in the integer math or array access in version 2. Incidentally, because of the use of a library in conjunction with a linking loader, the total object code generated by OMS for this program was under 2K. That is compared to 3.5K to 6K for most all of the others.

Since a few of the times didn't seem to agree very well with the Shearer article, (or to be more fair, with the times reported in BYTE) I decided to try all the others that I have. The first was TSC Extended BASIC. I coded the algorithm using integer variables. Since BASIC doesn't have the WHILE DO structure, I decided to code the main cross out loop as a FOR NEXT. An error showed up, and it turned out to be the case where the initial value of K% is larger than 8190, ie. 3!+3 > 8190. I had to add the test IF K>8190 THEN 160, to skip the FOR NEXT loop and only count the prime. The time for this program is 435 seconds compared to the 840 given in the article. It could be coded using GOTO, but the coding here more nearly matches the C and Pascal versions.

I decided to try TSC Pascal (FLEX9 version), and my results were 22.5 seconds compared to the reported 54! I suspect that result was at 1 MHZ also. The only other compiler in the Shearer list that I have is Dynasoft Pascal. I compiled the Pascal version in that also, and it ran in 255 seconds compared to the 309 in the article. You might be interested to know that the total runtime plus user program byte count for this package is 706! This is an integer only implementation, which is fine for many applications.

The variance between my results and those reported are too large to be a simple efficiency in my program over those reported. All four of the Pascal compilers I tested, were tested with the identical program. They all compiled it without errors. Yet my Dynasoft version ran in 82.5% of the reported time, while the same program in TSC Pascal ran in 42%. In the case of Lucidata, the number was 34%, and for TSC BASIC it was 52%. I should say here that my timing method was probably not as precise as that used in some of the tests, but the error seems large enough so that I think mine are a bit closer anyway. Late along here, I had an idea that possibly the new algorithm used in the January test reports had printed out the primes each time. I quickly decided that times like 8.78 seconds and 11 seconds would be impossible if that had been done.

LATE NOTE: I checked the BYTE article and found that I had printed the total prime count ten times whereas the benchmark algorithm printed it only once after ten iterations. Therefore, my times might err a bit on the long side. I am at a loss to explain the disparity between my times and those reported in BYTE, unless some of the reporters didn't take advantage of the features of the languages, such as using integer variables in the Extended BASIC version, etc.

By now, some of you are getting the idea that I am prejudiced. Of course I am! Comparsions are only fair if they present the best both sides have to offer. I'm trying to insure that at least one side presents its best case. Maybe I am a frustrated lawyer at heart.

COMPILER BENCHMARK TIMES

COMPILER	ANDERSON	SHEARER
IMS PASCAL	..	8.78
MC COSH C	10.0	....
INTROL C	..	11.0
PL/9	13.5	....
GSPL	18.0	....
OMEGASOFT PASCAL	20.7	40 **
TSC PASCAL FLEX9	22.5	54
IMS PCODE	...	105
BASIC09 INTERP	...	238
LUCIDATA PASCAL	250	735
DYNASOFT PASCAL	255	309
TSC X BASIC	435	840

```
NOTES: LUCIDATA, DYNASOFT, AND IMS PCODE ARE
       P-CODE IMPLEMENTATIONS.

       TSC X BASIC IS AN INTERPRETER.

       TESTS ON 2 MHZ FLEX9 SYSTEM.

       ** REPORTED IN BYTE 40 SEC ON IMHZ SYSTEM.
```

Some observations might be that three of "our" compilers are faster than "their" fastest on a 4 MHZ Z-80. We have six that run under 22 seconds. They have four that run 19 or under on a 4 MHZ Z-80. For either processor, the P-code interpreters tend to run about 10 times slower than the native code compilers. (IMS P-code seems to be an exception). I wonder if David Shearer will hear similar stories from the Z-80 users? For the record, I will include the listings of my benchmark coding for the Pascal, C, and BASIC versions here. Our Z-80 friends can look at them to see if I cheated anywhere. I consider using available data types such as CHAR in the Pascal version, and integer (I% etc.) in the BASIC version and 'char' or 'short' in the C version, to be making good use of the compiler and NOT cheating.

THE 6840

Now that I've praised Motorola for their fine 6809 design, I'd like to throw a brick through someone's window for ignoring the poor programmers in the design of the 6840 triple timer. Three timers, each of which require a 16 bit (word length) register to preset their count, and one control register, should need 9 memory addresses, right? Well some hardware designer decided that it would be better to "multiplex" the control registers for two of the timers into one address (access is via a bit in the third control register). I assume the reason was to save one register select and not "waste" 7 addresses of our very limited 65,536.

FOOEY! One more address bit (register select) would allow decoding 16 addresses. (Of course 7 of these are not needed). Squandering those extra seven bytes would save a lot more than that in the code necessary to initialize and use the timers in a typical application. Sure would make things a lot nicer in programming the 6840. Just think, that would mean there could be three identical timers with the timer registers addressed at XXX0, XXX4, and XXX8, and the control registers two addresses higher at XXX2, XXX6, and XXXA. Learn how to program one, and you have all three down pat. Too simple an idea for some hardware designer who doesn't have to use the timer to have thought of. Maybe they could even find uses for the other addresses. How about splitting functions into multiple registers to save some code in the inevitable ANDing and ORing that bit by bit programming forces upon the programmer. Those chips only have to be designed once. Manufacturing costs wouldn't change appreciably. Programmers would be forevermore grateful!

THE NAY HAS IT

I've never met Bob Nay, though I've talked to him on the phone a few times, so I don't feel like I'm "blowing our own horn" (or starting a mutual admiration society) when I say what follows. I read Bob's column every month with interest. In the June issue, he started a project with which I had been toying for some time. He started a series of columns on Programming a Computer in general, with no intention of getting very language specific.

I had started "test writing" a couple chapters or columns on that subject some time ago, but the project got bogged down. I found it extremely difficult not to get specific languages into the discussion very early. I decided that Bob was going to do a better job than I had the time, inclination (maybe talent) to do, and sent him my preliminary efforts. I hope he is able to glean a good idea or two from the material. Meanwhile I will be satisfied with writing random thoughts, projects, and test reports such as this.

```
/* Erathosthenes Sieve Prime Number Program in C */

#include stdio.h

#define true 1
#define false 0
#define size 8190
#define sizep1 8191

char flags[sizep1];

main()
   {int i, prime, k, count, iter;
   puts ("\n10 iterations\n");
   for (iter=1;iter<=10;++iter)
      {count=0;
      for (i=0;i<=size;++i)
         flags[i]=true;
      for (i=0;i<=size;++i)
         {if (flags[i])
            {prime=i+i+3;
            k=i+prime;
            while (k<=size)
               {flags[k]=false;
               k=k+prime;
               }
            ++count;
            }
         }
      }
   outdec (count);
   puts (" primes\n");
   }
outdec(n)
   int n;

   {int a;
   if (a=n/10) outdec (a);
   putchar (n%10+'0');
   }


 PROGRAM PRIME (INPUT,OUTPUT);

 { PRIME BENCHMARK PROGRAM IN PASCAL }

 CONST
    SIZE = 8190;

 VAR
    PRIME : ARRAY [0..SIZE] OF BOOLEAN;
   COUNT,I,J,K,L : INTEGER;

BEGIN
  FOR L := 0 TO 9 DO
  BEGIN
     COUNT := 0;
     FOR I := 0 TO SIZE DO PRIME [I] := TRUE;

     FOR I := 0 TO SIZE DO
     BEGIN
       IF PRIME [I] THEN
        BEGIN
          J:= I + I + 3;
          K := I+J;
          WHILE K <= SIZE DO
          BEGIN
             PRIME [K]:= FALSE;
             K := K+J;
          END;
          COUNT := COUNT + 1;
        END;
     END;
     WRITELN;
     WRITELN (COUNT:5,'PRIMES');
   END;
END.
```

```
30 DIM PR%(8190)
40 FOR L%=0 TO 9
50 CT%=0
60 FOR I%=0 TO 8190
70 PR%(I%)=1
80 NEXT I%
85 PRINT "DONE INIT"
90 FOR I%=0 TO 8190
100 IF PR%(I%)<>0 THEN 110 ELSE 170
110 J%=I%+I%+3
115 K%=I%+J%
119 IF K%>8190 THEN 160
120 FOR K%=K%TO 8190 STEP J%
130 PR%(K%)=0
140 NEXT K%
160 CT%=CT%+1
170 NEXT I%
180 PRINT
190 PRINT CT%;" PRIMES"
200 NEXT L%
210 END
```

OS9 USER NOTES

By: Peter Dibble
517 Goler House
Rochester, NY 14620

The OS-9 I/O System

OS-9 uses a modular I/O system designed for simplicity and flexibility. Because of this modularity an exceptionally ambitious user could write a new I/O subsystem and graft it into OS-9 without making any changes to the rest of the operating system. But there are other aspects of the I/O system which don't require any programming to exploit, and so useful that new OS-9 users should play with them as soon as possible.

The Unified Input/Output System

Each OS-9 process has three standard paths (files) open when it starts. Path 0 is called standard input, Path 1 is standard output, and path 2 is standard error. It is possible for a program to close these paths and re-open them for its own purposes, but most programs leave them open and use them as one might think they should be used.

The standard input path usually reads from the keyboard (terminal), and is used as the primary source of input from the user. Programs can and often do open other input files, sometimes the majority of the input is from some path other than standard input, but standard input is by convention the path used for communication with the user.

The standard output path typically writes to the screen (terminal), and is used for routine output to the user. Every character that appears on your screen probably came from a standard output path.

The standard error path is seldom used. By convention it is used for error messages. Normally the standard error path is directed to the screen together with the standard output path. The rationale for having separate paths for routine output and error messages rises from a special characteristic of the standard paths. Each of the paths can be directed wherever the user wishes before a program is started. This can prove useful when it is convenient to have different things done with error messages than with the rest of the output of a program.

The standard paths are open when a program starts because they are inherited from the process that started it, in most cases the shell. The shell takes advantage of this ability to pass its standard paths on to the programs it starts to change the paths from the standard (all to the terminal) to any other disposition a user might specify.

Options on a shell command line indicate to the shell what needs to be done to the standard paths. The options are ">xxxxx" for "redirect standard output to xxxxx," "<xxxxx" for "redirect standard input to xxxxx," and ">>xxxxx" for "redirect standard error to xxxxx." If any standard path is not redirected it is simply inherited from the shell; it usually goes to the terminal.

The ability to redirect the standard paths is called device independent I/O because paths can be directed to any device, not just another device of the same type as the default device for the path. The power of this feature is easiest to see with a few examples:

OS9: list filename is a command with no redirection. It lists the contents of the file called "filename" on the screen through the standard output path.

OS9: list filename >/P lists the contents of filename on the device called /P, usually the printer. The single ">" at the end of the command tells the shell to redirect the standard output to the file whose name follows the >. I can't think of any reason for someone to want to put the output of the list command into a disk file, but:

OS9: list filename >lstfile does just that. It puts the output of the list command into a file named lstfile. If you are using a multi-user system you can send the output of a command to another user with a command like:

OS9: asm test.a l >/T2 which would send the listing from the assembly of test.a to the device called /T2, which is usually a terminal.

I redirect Standard Output more than the other paths, but there are reasons to redirect the other paths as well. The Standard Input path is the one which programs usually read from. A program can be fed a canned script of commands by redirecting its Standard Input to a disk file with the commands in it. I sometimes insert this command in my startup file:

debug <startup.debug >/NL This runs the Microware debugger with its input coming from startup.debug, and its output going to a special SCF device which I made public in the first column I wrote (/NL is a null device -- it makes anything you send to it disappear). By putting debug in my startup file like this I can easily apply patches to resident modules every time I boot my system.

The Standard Error path is used so infrequently that it is easy to forget that it exists. It is the path which programs usually use for serious error messages. Usually, it is a good idea to leave the Standard Error path directed to the screen, but sometimes it should be redirected. Some compilers send syntax errors, or at least summary statistics out the Standard Error path. If you want to run a program that uses the Standard Error path in background while you edit in foreground, it is wise to redirect the both the Standard Output and the Standard Error paths of the compiler to disk files or the printer, otherwise you may find messages from the compiler cropping up in the middle of your screen at awkward times.

Redirection almost always works fine, but there are some problems lurking around. It shouldn't be the responsibility of a user to watch out for these problems, but OS-9 is designed with the assumption that programs will follow some conventions applying to their use of the standard paths. Some programs rely on dealing with particular devices. These programs should open special paths to those devices, but some use the standard paths for device dependent I/O. These programs should be avoided if possible.

The typical OS-9 system comes with three types of files, Sequential Character Files, Random Block Files, and Pipes. Sequential Character Files (usually called SCF files) are written or read from beginning to end. The most common SCF files are Terminal input and output, printer output, and modem input and output. The bytes in a RBF file (files handled by the RBFMAN file manager) can be read in any order. Disk files and other files like them, such as files in bubble memory or main memory, are usually RBF files. There is only one type of Pipe file, that is a temporary file kept in main memory which is used a buffer between one program's output and another program's input.

Unless a program concerns itself with timing issues or uses the more exotic GETSTAT/SETSTAT system service requests, there is no way for it to tell the difference between one device and another provided the devices are of the same type (RBF, SCF, or Pipe). Some programs can't have their standard I/O redirected to a RBF file or a Pipe, but the great majority can. If a program uses SCF-specific GETSTAT/SETSTAT codes it will only be possible to use it with the proper type of files, but all but one of the programs that I know of from Microware and other major vendors can have their I/O redirected without restriction. The one exception is Microware's Pascal with old versions of OS-9. All programs written in that language, including the compiler itself, try to rewind their standard output file when it starts. The SCF file manager deals with this strange request correctly by ignoring it, but the Pipe manager returns an error if anyone tries to rewind it. If you try to redirect the output of a program written in Pascal to a Pipe, the program will die as soon as it's started. Microware has a fix for this problem if you run into it.

Changing OS-9's Device Support

The modular design of OS-9's I/O system allows new devices to be added and the support of old devices to be enhanced with the only restrictions being the wishes and budget of the person responsible, and the memory constraints of the computer. Support for I/O starts at the IOMAN module which fields each I/O system service request and sometimes does a little work before passing it off to the appropriate module. File managers including SCF and RBF are the next level down from IOMAN; they do most of the file handling work that isn't specific to a particular piece of hardware. The device drivers, such as ACIA and PIA, handle the interface with the I/O hardware. The device descriptor modules contain the directions which all these modules follow. There is a descriptor for each device in an OS-9 system containing no executable instructions, but lots of data which controls the other I/O modules.

Hardware that requires complicated new modules for the I/O system should come with the necessary modules. The hardware vendor has to have the modules written (or write them), but a customer need only load the modules -- normally by including them in his boot -- in order to add software support for the device to his system. This sets OS-9 apart from many operating systems in which a major part of the operating system has to be changed for any new device.

Hardware vendors often need to write I/O modules in order to sell their products to the OS-9 community, but anyone can write I/O modules if the need or the mood takes them. Writing an entire new I/O subsystem would require a lot of work, but most problems can be solved with much less effort. Many devices can be accommodated by OS-9 without any serious programming at all by creating new device descriptors. Device descriptor modules specify how each device is to be treated. The device descriptor contains fields which indicate (to IOMAN) which file manager and device driver should be used for the device, an absolute physical address for the device, and any other data specific to the particular device

The first 18 bytes of all device descriptors have the same format. The first nine bytes are common to all module headers (Sync Bytes, Module size, Offset to Module Name, Type/Language ($F1), Attributes/Revisions and Header Parity check). Of these, the module attributes are most interesting in the context of the device descriptor. If the device descriptor module is marked reentrant, the device can be used by more than one process at a time; otherwise, it can only be linked to or opened by one process at a time. Device descriptors which are not reentrant are not only restricted to use by only one process at a time, they can't be linked to by debug at all if they are in the boot. Some devices, such as the printer, shouldn't be reentrant unless you feel very ready to be responsible. OS-9 will happily mix output from several programs line by line on the printer if you tell it to.

The format of the next nine bytes is common to all device descriptors. The fields are: the offset to the File Manager name (e.g., RBF) for two bytes, the offset to the Device Driver name (e.g., ACIA) for two bytes , the mode (what the device can do, e.g. Read/Write/execute) for one byte, the device controller's real address for three bytes, and the length of the initialization table.

After the first 18 bytes, different types of devices have different fields. The initialization table which follows the byte with its length contains most of the fields that are interesting to play with. After the initialization table there is nothing but module names and the CRC.

There are eleven fields in the initialization table for RBF-type devices (disk drives). The first field is one byte long and contains a 1 indicating that this is a RBF device. The other fields are:
drive number
step rate
device type
media density (0=single,1=double)
number of cylinders (two bytes long)
number of surfaces, verify (0=verify writes)
default sectors per track for two bytes
default sectors per track on track zero for two bytes
sector interleave factor
segment allocation size The step rate can take on values of 0..3 with the higher numbers reflecting higher stepping rates.

In the device type byte three bits are significant. Bit zero indicates a 8" floppy if it is one. Bit six indicates a non-standard format is being used if it is one. Bit seven being one indicates that the device is a hard disk.

In the media density byte two bits are significant. Bit zero = 1 indicates that the device can handle double density. Bit one = 1 indicates that the disk is capable of double track density (96 tpi).

The fields in the device descriptor are interpreted by the device driver and the file manager. Changing a value in the device descriptor can't force the other modules to do something they weren't written to do. For example, it probably isn't possible to use the device driver which is designed for floppy disks to control a hard disk -- changing the device type byte won't change the capabilities of the device driver. It is the option of the person writing the device driver to ignore anything in the device descriptor he wants. This means that there is no guarantee that the options in the device descriptor will work. I have heard that the floppy disk driver on the color computer ignores many of the options, I'll confirm this when I get one.

A different set of fields are in the initialization table for SCF devices. Most of these fields control the line-editing function of the SCF manager. These are the values that are temporarily set by TMODE. They can be set permanently by changing them in the device descriptor.

The initialization table in the device descriptor is copied into the path descriptor when a path is opened. There it can be changed and read by GETSTAT/SETSTAT calls, but the change applies only to that particular path. Changes to the device descriptor become the default for all paths opened to that device.

The easiest way to change the device descriptors is with debug. If, for example, you want to add a new terminal to your system which you don't have a device descriptor for, you can modify a similar descriptor with debug to fit your requirement (probably changing only the controller address and module name), save the result with the save command, and verify it with the update option to fix its CRC. The resulting module can be loaded and used.

A device descriptor can be modified even while the device it specifies is in use because the descriptor itself is seldom referenced. In fact, as far as I know, the device descriptor is only used when a path is opened to the device.

The device descriptor is the controlling part of the OS-9 I/O structure. There are several things that can be done with them that I haven't covered yet, but that will be material for other columns.

CONVERTING UNIFLEX TO OS9

The Conversion of Assembler-Language
Motorola 6809 Application Programs
From the UniFLEX Operating System
to the OS/9 Operating System

by E. M. (Bud) Pass, Ph.D.
Computer Systems Consultants, Inc.
1454 Latta Lane, Conyers GA 30207
Telephone Number 404-483-1717/4570

INTRODUCTION

A previous article in this same series provided a set of guidelines, procedures, and concepts for the conversion of assembler-language application programs written for the FLEX operating system to operate under the control of the OS/9 operating system.

This article extends the discussion to include the conversion of assembler-language programs written for the UniFLEX operating system to execute under the OS/9 operating system. It does not repeat most of the information presented in the earlier article concerning OS/9. Thus, it may be necessary for the reader unfamiliar with the details of assembler-language programming in OS/9 to refer to that article or other information to be able to understand some of the concepts presented in this article.

The extent of the differences between UniFLEX and OS/9 is far greater than the extent of the differences between FLEX and OS/9, even though they both offer facilities externally similar to UNIX. This is true because of the complex natures of both OS/9 and UniFLEX. Thus, the suggested conversions will be less specific, in general, than those suggested for the conversion from FLEX to OS/9, and the manual conversions required may be somewhat more difficult.

UniFLEX BACKGROUND

UniFLEX is one of several products of Technical

Systems Consultants, Inc. Another product of TSC is the FLEX operating system discussed in the previous article in the series.

UniFLEX is a operating system for Motorola 6809- and 68000-based computer systems. It supports full-size floppies and hard disks, with tree-structured directories on each drive, along with CRT terminals and printers. It is reliable, well-supported, and easy-to-use.

UniFLEX supports up to 1M of RAM and 1G bytes per drive of disk. UniFLEX is inherently multi-user and multi-tasking. It requires a minimum of 96K (realistically 128K-256K) for normal operations.

UniFLEX COMMAND LINE

The UniFLEX command line is formatted as follows:

```
command [p1]...[pn] [<fnl] [[>]>fno] [^pipe] [&] [;...]
```

where: command is the name of the program to be executed;

p1...pn are command parameters;

<fin is a redirected standard input path;

[>]>fno is a redirected standard output path;
">>" indicates deletion of existing file;

^pipe is an indicator specifying pipeline processing;

& is an indicator specifying background processing;

[;...] is multiple commands, executed left to right.

Parentheses may be used to force alternate grouping of commands.

UniFLEX FILE NAMES

A UniFLEX file name has the following format:

```
[/dir1/.../dirn/]filename
```

where: /dir1/.../dirn/ is an optional designation of the file's location in the UniFLEX tree-structured directory;
filename is a 1-14 character file name, starting with a letter.

File names and directory names used in operands may also contain character sequences such as "*", "?", "[...]", acting as wild-card designates to specify (possibly) multiple files without providing complete names. UniFLEX expands these ambiguous file names to lists of real file names before passing the command line to the program being invoked.

Both commands and file-names have the default position in the UniFLEX tree-structured directory specified at user log-on or by the last "chd" command. Commands not found in the working directory and with names not starting with a slash are also looked for in the "/bin" and "/usr/bin" directories.

UniFLEX MEDIA CONVERSION PROBLEMS

Since UniFLEX uses a disk format incompatible with both OS/9 and FLEX, direct media conversion from UniFLEX to OS/9 is not possible.

Source-program and data files could be transferred over modems and telephone lines or over direct-connect serial lines.

The 'writeflex' UniFLEX utility command could also be used to convert the source-program and data files to FLEX format, and the 'of' program available from DATA-COMP could be used to convert the files from FLEX format to OS/9 format. No FLEX system is required, except to format the diskettes used in the conversion process. Of course, the diskettes must be of compatible physical media for this process to work without the use of another system.

UniFLEX PROGRAM INTERFACE SUMMARY

UniFLEX communicates with application programs in a manner somewhat similar to that used by OS/9: each request is a system call (swi3 on the 6809, trap on the 68000), followed by a list of parameters.

Each application program has access to the full 64K byte of RAM address space, with program code normally loaded from the lower addresses and stack normally advancing from the higher addresses, each in 4K byte blocks. Unallocated memory between the two limits is not accessible by programs, and neither are the I/O and DAT addresses.

An application program may spawn additional tasks and specify that they are to run in independent, concurrent, or pipelined mode, and, in the latter two modes, monitor their progress and completion status.

In case the demands on the system exceed the available memory space, UniFLEX will automatically initiate task-swapping to the system residence device. Since UniFLEX programs are assumed non-position-independent, they are always reloaded in the original logical address space.

When any task is initiated by UniFLEX, the command-line parameters are passed in the stack, as represented by the following diagram:

```
      parameter terminator        $0000 (higher addresses)
         :
      parameter 2 terminator      $00
      parameter 2                 '....' <==
      parameter 1 terminator      $00       |
      parameter 1                 '....' <=|====
      parameter 0 terminator      $00       |    |
      parameter 0                 '....' <=|====|=====
      parameter list terminator   $0000     |    |    |
         :                                  |    |    |
      parameter 2 pointer         $xxxx  ====    |    |
      parameter 1 pointer         $xxxx  =========    |
      parameter 0 pointer         $xxxx  ==============
sp==> parameter count             $nnnn  (lower addresses)
```

Note that parameter zero in the diagram above is the name of the program being initiated.

The format for a UniFLEX system call in UniFLEX assembler language is as follows:

```
sys    code[,para1,...,paran]
```

where: code is one of the system call functions described below:
it is interpreted as an 8-bit call code;

para1 thru paran are call parameters;
they are interpreted as 16-bit values.

In addition, parameters are passed between UniFLEX and the application program in the CC, D, U, X, or Y registers. In general, a system call of the following format:

```
sys  code,para1,...,paran
```

would be equivalent to the following code:

```
swi3
fcb    code
fdb    para1
  :
fdb    paran
```

where code and para1 thru paran must be constants or given values elsewhere in the assembler program containing the UniFLEX call. The number of parameters for a given system call is always fixed, but the assembler does not check for the correct number of arguments.

Since UniFLEX has no method of checking for the correct number of arguments for a given system call, it will return to the program at an offset specified by the system call code, not necessarily the actual number, with potentially disastrous results, in case of error.

The UniFLEX assembler supports the specification of segmented code sequences. The text, data, and bss pseudo-operations are used for this purpose, as they are under UNIX.

The text pseudo-operation specifies the generation of (read-only) shared-text program-code and constant-data areas which may be shared by multiple simultaneous users of the same program.

The data pseudo-operation specifies the generation of (read-write) initialized data areas. If a data segment is used by multiple simultaneous users, each has a separate copy. A data segment must start at the next 4K-byte address boundary after the end of the text segment.

The bss pseudo-operation reserves memory but does not initialize it, serving a function similar to the data pseudo-operation.

The base pseudo-operation provides a temporary secondary program counter origin (like org) which may be used to help establish offsets and pointers but may not be used to generate programs or data code.

UniFLEX uses a 512-byte disk sector format, with all 512 bytes available for data storage. File linkage and allocation information is maintained externally to the user-accessible data records, removing the potential opportunity for destruction of file linkage information.

UniFLEX FACILITIES

Following is a list of the primary UniFLEX system calls, their code numbers, and their descriptions:

```
Code  System Call   Description

00 ind(address)
        Perform an indirect system call
01 indx()
        Perform an indexed system call
02 exec(pathname,arglist)
        Execute a new program
03 fork()
        Fork a new task
04 wait()
        Wait for a forked task to complete
05 term(status)
        Terminate a task
06 break(address)
        Adjust the data memory size
07 stack(address)
        Adjust the stack memory size
08 cpint(interruptnumber,address)
        Catch program interrupt
09 splnt(tasknumber,interruptnumber)
        Send a program interrupt
10 open(pathname,mode)
        Open a file for read or write or both
11 create(pathname,permission)
        Create and open an output file
12 read(filenumber,address,count)
        Read from a file
13 write(filenumber,address,count)
        Write to a file
14 seek(filenumber,offsethi,offsetlo,type)
        Position file pointer
15 close(filenumber)
        Close a file
16 dup(filenumber)
        Duplicate an open file
17 dups(filenumber,specifiednumber)
        Duplicate a file
18 link(oldpathname,newpathname)
        Link to an existing file
19 unlink(pathname)
        Unlink from a file (delete)
20 crtsd(pathname,desc,address)
        Create a directory, etc.
21 chdir(pathname)
        Change working directory
22 lock(mode)
        Lock a task in main memory
23 chown(pathname,newowner)
        Change file owner
24 chprm(pathname,permission)
        Change file access permission
25 chacc(pathname,permission)
        Check file access permission
26 defacc(permission)
        Set default file access permission
27 ofstat(filenumber,buffer)
        Get an opened file's status
28 status(pathname,buffer)
        Get a file's status
29 mount(devicepathname,pathname,mode)
        Mount a device
30 unmnt(devicepathname)
        Unmount a device
31 crpipe()
        Create an inter-task pipe
32 gtid()
        Get the task id
33 guid()
        Get the user id
34 suid(newuserid)
        Set the user id
35 setpr(priority)
        Set running priority bias
36 cdata(address)
        Allocate physically contiguous memory
37 profil(pc,buffer,size,scale)
        Profile a running task
38 trap(address)
        Set swi2 trap vector
39 time(buffer)
        Get the time
40 stime(timehi,timelo)
        Set the time
41 ttime(buffer)
        Get tasks's system time information
42 update()
        Update all system devices
43 alarm(seconds)
        Wait for a specified time period
44 stop()
        Stop task until interrupt
45 ttyget(filenumber,address)
        Get terminal status and info
46 ttyset(filenumber,address)
        Set terminal configuration
47 lrec(filenumber,size)
        Lock the specified file's bytes
48 urec(filenumber)
        Unlock the record
50 sacct(fname)
        System accounting
51 ttynum()
        Get TTY number
52 filtim(fname)
        Set file time
```

Following are the UniFLEX system call and control block definitions:

```
Number Name   Description

        * Uniflex system calls

$00   ind     indirect call
$01   indx    index indirect call
$02   exec    exec
$03   fork    fork
$04   wait    wait
$05   term    terminate task
$06   break   exetend memory address
$07   stack   grow stack
$08   cpint   catch program interrupt
$09   spint   send program interrupt
```

```
$0A  open    open file
$0B  create  create a file
$0C  read    read file
$0D  write   write file
$0E  seek    seek to file position
$0F  close   close file
$10  dup     duplicate open file
$11  dups    duplicate specific file
$12  link    link to file
$13  unlink  unlink from file
$14  crtsd   make special or directory file
$15  chdir   change directory
$16  lock    lock task in memory
$17  chown   change file owner
$18  chprm   change access perm
$19  chacc   check access permission
$1A  defacc  set default access
$1B  ofstat  get open file status
$1C  status  get file status
$1D  mount   mount device
$1E  ummnt   unmount device
$1F  crpipe  create pipe
$20  gtid    get task id
$21  guid    get user id
$22  suid    set user id
$23  setpr   set priority bias
$24  cdata   request contiguous data
$25  profil  profile task
$26  trap    set swi2 trap vector
$27  time    get time
$28  stime   set time
$29  ttime   get task time
$2A  update  update file systems
$2B  alarm   sleep for some seconds
$2C  stop    stop til interrupt
$2D  ttyget  get tty status
$2E  ttyset  set tty status
$2F  lrec    lock file record
$30  urec    unlock file record
$32  sacct   system accounting
$33  ttynum  get tty number
$34  filtim  set file access time

        * System error definitions

$01  EIO     io error
$02  EFAULT  system fault
$03  EDTOF   data section overflow
$04  ENDR    not a directory
$05  EDFUL   disk full
$06  ETMFL   too many files
$07  EBADF   bad file
$08  ENOFL   no file
$09  EMSDR   missing directory
$0A  EPRM    file permission
$0B  EFLX    file exists
$0C  EBARG   bad argument
$0D  ESEEK   seek error
$0E  EXDEV   crossed devices
$0F  ENBLK   not a block special file
$10  EBSY    device is busy
$11  ENMNT   file not mounted
$12  EBDEV   bad device specified
$13  EARGC   too many arguments
$14  EISDR   file is a directory
$15  ENOTB   file not binary
$16  EBBIG   binary file too big
$17  ESTOF   stack overflow
$18  ENCHD   no children living
$19  ETMTB   too many tasks active
$1A  EBDCL   bad system call
$1B  EINTR   interrupted system call
$1C  ENTSK   no task found
$1D  ENTTY   not a tty
$1E  EPIPE   write to broken pipe
$1F  ELOCK   record locking error

        * system interrupt definitions

$01  HANGI   hangup interrupt
```

```
$02  INTI    keyboard interrupt
$03  QUITI   quit interrupt
$04  EMTI    emt interrupt (swi)
$05  KILLI   kill task interrupt
$06  WPIPI   write broken pipe interrupt
$07  BARGI   bad argument interrupt
$08  TRACI   trace interrupt
$09  TIMEI   time limit interrupt
$0A  ALRMI   alarm interrupt
$0B  TERMI   termination interrupt
$0C  USERI   user-defined interrupt

        * file status block structure definition

$00  SZ_TMP  save current pc
$00  st_dev  device number
$02  st_fdn  fdn number
$04  st_mod  file modes
$05  st_prm  permission bits
$06  st_cnt  file link count
$07  st_own  file owner's user id
$09  st_siz  file size in bytes
$0D  st_mtm  last modified time
$11  st_spr  spare - future use only
$15  ST_SIZ  size of status buffer

        * modes

$02  FSBLK   block device
$04  FSCHR   character device
$08  FSDIR   directory

        * permissions

$01  FACUR   owner read permission
$02  FACUW   owner write permission
$04  FACUE   owner execute permission
$08  FACOR   others read permission
$10  FACOW   others write permission
$20  FACOE   others execute permission
$40  FXSET   set id bit for execution

        * 'time' buffer

$00  tm_sec  time in seconds
$04  tm_tik  ticks in second (tenths)
$05  tm_zon  time zone
$07  tm_dst  daylight savings flag
$08  TM_SIZ  time buffer size

        * 'ttime' buffer

$00  ti_usr  task's user time
$03  ti_sys  task's system time
$06  ti_chu  children's user time
$0A  ti_cha  children's system time
$0E  TI_SIZ  size of ttime buffer

        * buffer definitions for 'ttyset' and 'ttyget'

$00  tt_flg  flags
$01  tt_dly  delays
$02  tt_cnc  line cancel character
$03  tt_bks  backspace character
$04  tt_spd  terminal speed (not used)
$05  tt_spr  spare - future use only
$06  TT_SIZ  size of tty buffer

        * flags

$01  RAW     raw i/o mode
$02  ECHO    echo input characters
$04  XTABS   expand tabs on output
$08  LCASE   map upper to lower case & vice versa
$10  CRMOD   output cr and lf for cr
$20  BSECH   output backspace echo character
$40  SCHR    single character input mode
$80  CNTRL   ignore control characters
```

The UniFLEX system calls and corresponding control block descriptions are always found within a UniFLEX assembler language program. Most of the considerations discussed for the conversion of FLEX programs to OS/9 also apply to the conversion of UniFLEX programs, in addition to some other considerations not already discussed.

Of course, each OS/9 executable module must contair a module header and trailer, possess position-independent code and data, use 6809, not 6800 mnemonics, etc., as discussed for FLEX programs.

Many UniFLEX programs have additional assembler-language constructs which must also be converted to OS/9 assembler-language. The simplest are the names of the common source libraries such as "sysdef", which are stored in the "/lib" directory. These libraries provide standard symbolic labels for the UniFLEX system call codes and control blocks.

UniFLEX supports segmented programs, which involve the use of shared program code. However, the programmer must designate to the assembler which portions of the program are shared code, initialized data, and uninitialized data. This is done with the text, data, and bss pseudo-opcodes. Since OS/9 automatically shares program text, the text pseudo-opcode may be safely dropped. Since OS/9 does not initialize the assigned data space, the bss portion may be used directly, but must be combined with the data portion, if any, and program code must be added to copy the former initialized data areas into the OS/9 uninitialized data space.

OS/9 IMPLEMENTATION OF UniFLEX SYSTEM CALLS

This section discusses the conversion of the UniFLEX system calls and control blocks to OS/9 format. Almost all of the commonly-used UniFLEX system calls have corresponding OS/9 system calls, and almost all of the commonly-used UniFLEX control blocks have their corresponding OS/9 control blocks. However, "corresponding" does not mean "identical". There are no one-to-one correspondences among any of the UniFLEX and OS/9 control blocks, and there are very few one-to-one correspondences among the system calls.

The error and interrupt number codes have different meanings and values on the two systems; however, most of the differences are simple to resolve with an editor or with a cross-reference equate file, providing the equivalent OS/9 values for the UniFLEX symbol names. Both systems use the carry bit set to indicate an error condition during the invocation of a system call. UniFLEX uses the D register to return the error number, whereas OS/9 uses the B register; even though the error numbers are different, they will still be in the B register, and the same values will usually be in the D register, since the A register is cleared under most of the suggested conversions.

Following are the UniFLEX system calls and suggested OS/9 conversions. As in the case of FLEX, many of the call conversions may be grouped together into a library, or they may be placed at the point of each call. If they are grouped, some of the suggested conversions must be modified, as they use parameters not in registers.

System Number and Call
    Description
        OS/9 Conversion

00 ind(address)
    Perform an indirect system call
        This system call has no direct OS/9 equivalent conversion without the use of an interpreter; usually, another specific UniFLEX system call is being made and may be substituted for the indirect system call.

01 indx()
    Perform an indexed system call
        This system call has no direct OS/9 equivalent conversion without the use of an interpreter; usually, another specific UniFLEX system call is being made and may be substituted for the indexed system call.

02 exec(pathname,arglist)
    Execute a new program
        UniFLEX arguments are indirectly referenced thru a list of addresses in the stack, all terminated with a hex zero address, and each argument is terminated with a hex zero byte; OS/9 requires the parameters to be a string of characters, separated by spaces, and terminated with a carriage return.

```
        pshs x,y,u
        leay $08,s   command file name
        ldb  #$20
paraadr ldx  ,y++
        beq  paraend
paralop lda  ,x+     parameter terminator
        bne  paralop
        stb  -$01,x
        tfr  x,u
        bra  paraadr
paraend ldb  #$0d    parameter area end
        stb  ,u
        pshs u
        ldd  $08,s
        subd ,s++
        tfr  d,y     parameter size
        leau $0a,s   parameter area
        ldx  $08;s   program name
        clra         language/type code
        clrb         data pages
        os9  F$Chain
        puls x,y,u
```

03 fork()
    Fork a new task
        UniFLEX starts a new task with the fork system call by copying the caller's program image to a new area, then initiating both the old and new tasks, with the old task returning two bytes past the end of the system call with the child's task id in the D register; OS/9 starts new tasks with the F$Fork system call, which has parameters as described for the F$Chain system call.

04 wait()
    Wait for a forked task to complete

```
        pshs x
        clra
        os9  F$Wait
        bcs  waiterr
        tfr  d,x
        clra
        exg  d,x
        tfr  a,b
        clra
waiterr puls x
```

05 term(status)
    Terminate a task

```
        os9  F$Exit
```

06 break(address)
    Adjust the data memory size
        UniFLEX allocates memory very differently from OS/9 Level I and Level II; the closest OS/9 system call is F$Mem, which allocates memory toward higher addresses above the stack, whereas UniFLEX allocates additional data space above the program and data space, toward the stack.

07 stack(address)
    Adjust the stack memory size
        UniFLEX allocates memory very differently from OS/9 Level I and Level II; the closest OS/9 system call is F$Mem, which allocates memory toward higher addresses above the stack, whereas UniFLEX allocates additional stack space toward lower addresses, below the stack.

08 cpint(interruptnumber,address)
    Catch program interrupt
        UniFLEX program interrupt handling is quite different from OS/9 program interrupt handling; the corresponding OS/9 system call is F$Icpt,

which handles all program interrupts once established, whereas the UniFLEX system call handles only one class of program interrupts per invocation; both return with "rti".

09 spint(tasknumber,interruptnumber)

Send a program interrupt

UniFLEX and OS/9 interrupt codes are different, as noted earlier.

```
         tfr  b,a                task number
         ldb  #interruptnumber signal code
         os9  F$Send
         lda  #$00
```

10 open(pathname,mode)

Open a file for read or write or both

UniFLEX file names are delimited by trailing hex zero bytes; OS/9 file names are normally delimited by a high-order-bit set on the last byte of the file name; however, OS/9 seems to accept the UniFLEX convention, probably since the UniFLEX convention is identical to that used by the C compilers under both systems.

```
         pshs x
         lda  #(mode+1)
         tfr  a,b
         lslb
         lslb
         lslb
         pshs b
         ora  ,s+
         leax pathname,pcr (or ,u ...)
         os9  I$Open
         bcc  openok
         tfr  a,b
openok   lda  #$00
         puls x
```

11 create(pathname,permission)

Create and open an output file

UniFLEX file names are delimited by trailing hex zero bytes; OS/9 file names are normally delimited by a high-order-bit set on the last byte of the file name; however, OS/9 seems to accept the UniFLEX convention, probably since the UniFLEX convention is identical to that used by the C compilers under both systems; UniFLEX automatically deletes already-existing files, but OS/9 does not do so automatically.

```
         pshs x
         ldb  #permission
         tfr  b,a
         lda  #$03         read/write
         leax pathname,pcr (or ,u ...)
         os9  I$Create
         bcc  creatok
         tfr  a,b
creatok  lda  #$00
         puls x
```

12 read(filenumber,address,count)

Read from a file

UniFLEX allows the program to partially control the actions of the read system call thru certain bits in the flag byte paramete((tt_flg); the input options controlled by this parameter are: raw I/O, echo input, map case, echo backspace, single character, ignore control characters; OS/9 controls raw I/O with I$Read or I$ReadLn calls and some of the other options with path descriptor parameters; UniFLEX returns zero bytes and no error indication on end of file, whereas OS/9 returns E$EOF error code.

```
         pshs x,y
         tfr  y,d
         tfr  b,a
         ldy  #count
         leax address,u   (or ,s ...)
         os9  I$Read      (or I$ReadLn)
         lda  #$00
         bcs  readok
         tfr  y,d
readok   puls x,y
```

13 write(filenumber,address,count)

Write to a file

UniFLEX allows the program to partially control the actions of the write system call thru certain bits in the flag byte parameter (tt_flg); the output options controlled by this parameter are: raw I/O, expand tabs, map case, auto line feed; OS/9 controls raw I/O with I$Read or I$ReadLn calls and some of the other options with path descriptor parameters.

```
         pshs x,y
         tfr  y,d
         tfr  b,a
         ldy  #count
         leax address,pcr (or ,u ...)
         os9  I$Write     (or I$WritLn)
         lda  #$00
         bcs  writok
         tfr  y,d
writok   pula x,y
```

14 seek(filenumber,offsethi,offsetlo,type)

Position file pointer

UniFLEX has the following three base points for the seek call: beginning of file, current position, end of file, and returns the resultant position in the file; the OS/9 seek call is always from the beginning of the file, and the getstat call is used to determine the position within the file.

```
         pshs d,x,y,u
         tfr  u,y
         ldb  #type
         beq  seek0
         cmpb #$01
         beq  seek1
         cmpb #$02
         beq  seek2
seek0    ldx  #$0000       beginning
         ldu  #$0000
         bra  seekit
seek1    ldb  #$05         current
         os9  I$GetStt
         bcs  seekerr
         bra  seekit
seek2    ldb  #$02         end of file
         os9  I$Getstt
         bcs  seekerr
seekit   tfr  u,d
         addd #offsetlo
         tfr  d,u
         pshs x
         ldd  #offsethi
         adcb $01;s
         adca ,s++
         tfr  d,x
         lda  $01,s
         os9  I$Seek
         bcs  seekerr
         ldb  #$05
         os9  I$GetStt
         bcs  seekerr
         stx  $02,s
         stu  ,s
         bra  seekex
seekerr  lda  #$00        seek error
         std  ,s
seekex   puls d,x,y,u
```

15 close(filenumber)

Close a file

```
         tfr  b,a
         os9  I$Close
         lda  #$00
```

16 dup(filenumber)

Duplicate an open file

```
         tfr  b,a
         os9  I$Dup
         lda  #$00
```

17 dups(filenumber,specifiednumber)

Duplicate a file

The nearest OS/9 equivalent does not allow the calling program to specify the file descriptor of the newly created file.

```
        tfr  b,a
        os9  I$Dup
        lda  #$00
```

18 link(oldpathname,newpathname)

Link to an existing file

UniFLEX supports the concept of a link file, which is not an actual file, but provides a pointer to another file, which is then used as if it were the link file; OS/9 does not support such a concept.

19 unlink(pathname)

Unlink from a file (delete)

UniFLEX has only the working directory; OS/9 has both the data and execution directories, and allows the deletion of files from either.

```
        pshs x
        leax pathname,pcr (or ,u ...)
        tfr  b,a
        os9  I$Delete     (or I$DeletX)
        lda  #$00
        puls x
```

20 crtad(pathname,desc,address)

Create a directory, etc.

UniFLEX allows the creation of block or character devices or of directories with this system call; OS/9 allows only the creation of directories with the closest equivalent system call.

```
        pshs x
        leax pathname,pcr (or ,u ...)
        ldd  #desc
        cmpa #$08         make sure directory
        beq  crtad
        orcc #$01         set error
        ldb  #$E$UnkSvc
        bra  crtader
crtad   andb #$3f         reset id bit
        os9  I$MakDir
crtader lda  #$00
        puls x
```

21 chdir(pathname)

Change working directory

UniFLEX has only the working directory; OS/9 has both the data and execution directories.

```
        pshs x
        leax pathname,pcr (or ,u ...)
        lda  #$03   (or $04 for execution dir)
        os9  I$Chgdir
        lda  #$00
        puls x
```

22 lock(mode)

Lock a task in main memory

OS/9 has no equivalent to this system call.

23 chown(pathname,newowner)

Change file owner

OS/9 has no direct equivalent to this system call.

24 chprm(pathname,permission)

Change file access permission

OS/9 has no direct equivalent to this system call; in OS/9, the 'attr' command may be used to change the file's access permissions.

25 chacc(pathname,permission)

Check file access permission

OS/9 has no direct equivalent to this system call; in OS/9, the 'attr' command may be used to determine the file's access permissions.

26 defacc(permission)

Set default file access permission

OS/9 has no direct equivalent to this system call; in OS/9, children tasks inherit the open files of the parent task, so default file access permissions are meaningless.

27 ofstat(filenumber,buffer)

Get an opened file's status

OS/9 has no direct equivalent to this system call; in OS/9, the 'attr' command may be used to determine the file's access permissions, and the I$GetStt system call may be used to determine file size and other file parameters.

28 status(pathname,buffer)

Get a file's status

OS/9 has no direct equivalent to this system call; in OS/9, the 'attr' command may be used to determine the file's access permissions, and the I$GetStt system call may be used to determine file size and other file parameters.

29 mount(devicepathname,pathname,mode)

Mount a device

OS/9 has no direct equivalent to this system call; in OS/9, devices are always designated by the device name, not by an indirect directory name.

30 unmnt(devicepathname)

Unmount a device

OS/9 has no direct equivalent to this system call; in OS/9, devices are always designated by the device name, not by an indirect directory name.

31 crpipe()

Create an inter-task pipe

OS/9 has no direct equivalent to this system call; in OS/9, pipes are created by the shell process, not within application programs.

32 gtid()

Get the task id

```
        pshs y
        os9  F$ID
        bcc  gtidok
        tfr  a,b
gtidok  lda  #$00
        puls y
```

33 guid()

Get the user id

```
        pshs y
        os9  F$ID
        bcc  guidok
        tfr  y,x
        tfr  y,d
guidok  lda  #$00
        puls y
```

34 suid(newuserid)

Set the user id

Only OS/9 Level II allows the F$SUser system call.

```
        pshs y
        clra
        tfr  d,y
        os9  F$SUser
        lda  #$00
        puls y
```

35 setpr(priority)

Set running priority bias

The UniFLEX system call setpr specifies a bias value; the OS/9 system call F$Sprior specifies the process priority value directly.

```
        pshs x,y
        tfr  a,b
        clra
        tfr  d,x
        os9  F$ID
        bcs  setprer
        pshs a
        tfr  x,d
        puls a
        os9  F$Sprior
setprer lda  #$00
        puls x,y
```

36 cdata(address)

Allocate physically contiguous memory

UniFLEX allocates memory very differently from OS/9 *Level* I and *Level* II; the closest OS/9 system call is F$Mem, which allocates memory toward higher addresses above the stack, whereas UniFLEX allocates additional data space above the program and data space, toward the stack.

37 profil(pc,buffer,size,scale)
    Profile a running task
        OS/9 has no equivalent to this system call.

38 trap(address)
    Set swi2 trap vector
        Since programs running under control of OS/9 use swi2 for system calls, swi2 may not be used for other purposes by programs; however, swi or swi3 may be used; the OS/9 system call does not return the previous swi or swi3 address.

```
                pshs x
                lda  #$01        (or $03 for swi3)
                leax address,pcr (or ,u ...)
                os9  F$SSWI
                lda  #$00
                pula x
```

39 time(buffer)
    Get the time
        UniFLEX returns the time in seconds and ticks from 0000 January 1, 1980 UTC; OS/9 returns the time in years, months, days, hours, minutes, and seconds from 0000 January 1, 1900 UTC, in a buffer, one byte for each item.

```
                pshs x
                leax buffer,u    (or ,s ...)
                os9  F$Time
                lda  #$00
                pula x
```

40 stime(timehi,timelo)
    Set the time
        UniFLEX requires the time in seconds in X and D from 0000 January 1, 1980 UTC; OS/9 requires the time in years, months, days, hours, minutes, and seconds from 0000 January 1, 1900 UTC, in a buffer, one byte for each item.

```
                pshs x
                leax buffer,u    (or ,s ...)
                os9  F$STime
                lda  #$00
                puls x
```

41 ttime(buffer)
    Get tasks's system time information
        OS/9 has no equivalent to this system call.

42 update()
    Update all system devices
        OS/9 has no equivalent to this system call.

43 alarm(seconds)
    Wait for a specified time period
        OS/9 has no equivalent to this system call; an independent task could be generated which, after a specified number of seconds, generates an interrupt to the calling task and terminates.

44 stop()
    Stop task until interrupt

```
                pshs x
                ldx  #$0000
                os9  F$Sleep
                lda  #$00
                pula x
```

45 ttyget(filenumber,address)
    Get terminal status and info
        UniFLEX returns terminal status and information in a six-byte area called ttbuf; OS/9 returns some of the terminal option information in a 32-byte area which is a copy of the option section of the path descriptor and it returns the terminal status flag as a separate byte, with separate calls to the I$GetStt system call.

46 ttyset(filenumber,address)
    Set terminal configuration
        UniFLEX requires terminal status and information in a six-byte area called ttbuf; OS/9 requires some of the terminal option information in a 32-byte area which is copied into the option section of the path descriptor with a call to the I$SetStt system call.

47 lrec(filenumber,size)
    Lock the specified file's bytes
        OS/9 Level II locks the last record read in a file opened for update; this method is not directly compatible with the method used by UniFLEX, which allows the user to specify the range of bytes within a file currently being locked; both systems allow only one active lock per file.

48 urec(filenumber)
    Unlock the record
        OS/9 Level II locks the last record read in a file opened for update; this method is not directly compatible with the method used by UniFLEX, which allows the user to specify the range of bytes within a file currently being locked; both systems allow only one active lock per file.

50 sacct(fname)
    System accounting
        OS/9 has no equivalent to this system call.

51 ttynum()
    Get TTY number
        OS/9 has no equivalent to this system call.

52 filtim(fname)
    Set file time
        OS/9 has no equivalent to this system call.

SUMMARY

This article, like the first in the series, has attempted to provide a framework for the conversion of UniFLEX assembler application programs to operate under the OS/9 operating system.

It discussed the different and additional requirements placed on programs which run under OS/9, and suggested equivalent OS/9 system calls for many UniFLEX system calls.

Although UniFLEX and OS/9 are externally similar operating systems, they are internally quite different and are both very complex. Therefore, the suggested conversions from UniFLEX are far more complicated and incomplete than they were from OS/9 in the earlier article in this series.

With the information provided in the two articles in this series, a programmer familiar with assembler language for OS/9 and FLEX or UniFLEX should be able to convert assembler language application programs from FLEX or UniFLEX to OS/9.

1983 INDEX
68 MICRO JOURNAL
ARTICLES

The following is an index of articles and other mate ial published during the year 1983. Some back issues are still available for $3.95 each plus mailing and handling.

January

FLEX User Notes, Anderson
COLOR User Notes, Nay
HEX +, Baleshta
C User Notes, Commo
Dual Processors, Green
OS9-Eprom, Trollope
UNIFLEX FORTRAN Utilities, Matheny
PL9-MACE, Review
OSM Assembler, Review
Map-Color, Stearns
Modem & Hex, Antikainen & Kask
FLEX-SI, Everett
MEM EXM + Charge, Pladsen
Bit Bucket - Help - Classifieds

February

FLEX User otes, Anderson
COLOR User Notes, Nay
Color-IBM, Perotti
Color-Expansion Review, DMW
1982 Index
Label OS9, Accettola
Rumors & Such, DMW
QED Review, DMW
PB4 Buffer Review, DMW
Auto-Comm Review, DMW
A Perfect Number, Brakefield
DMS Notes, Adams
Color Graphics, Hunt
Fortran Utilities, Matheny
MEX68KECB (68000), Gallagher
Bit Bucket - Help - Classifieds

March

FLEX User Notes, Anderson
RUMORS & SUCH, DMW
Color User Notes, Nay
PENGUIN Sys, Pass
"C" User Notes, Commo
FLEXIBLE P SYS, Drexler
MODEM, Antikainen
68000 Dual Ports, West
FLEX Disk Name Change, Hoglund
PL-9 Update, Anderson
Bit Bucket - Help - Classifieds

April

FLEX User Notes, Anderson
COLOR User Notes, Nay
"C" User Notes, Commo
OS9 User Notes, Dibble
COPYMULT & OF, Review
Fast TRIG & MATH, Scudiere
Experimenting w/FLEX, Stark
PASCAL-FLEX Modules, Dean
BASIC's "User", Baker
CHECKERS, Bollinger
CoCo Disk 2 Sides, Meyers
HEX/DEC, Watson
UniFLEX Tips, Frank
Stylo Fix, Unkn
Home Acct-Update, Watson
UniFLEX-FORTRAN Routines, Mathney
Bit Bucket - Help - Classifieds

May

FLEX USER NOTES, Anderson
COLOR USER NOTES, Nay
'C' USER NOTES, Commo
'C' FLOATING POINT, Battelnger & Harkness
OS9 USER NOTES, Dibble
OS9 MAXIMUM EFFICIENCY, Burega
MICROWARE ANNUAL USER SEMINAR
STRUCTURED ASSEMBLE MACROS, Leclerc
SMALL/SIMPLE EXEC-68KECB-68000, Duclos
REGULATOR PROBLEMS, Stark
Editor's Note - Software copy, DMW
MODEM PROGRAM (another), Burega
TSC EDITOR MOD, Hughes
PTR-3 REVIEW, Thomas
Bit Bucket - Help - Classifieds

June

FLEX USER NOTES, Anderson
OS9 USER NOTES, Dibble
DY ACALC (Review - OS9), Dibble
O-F (Review -OS9 to FLEX to OS9), Dibble
A/D CONVERTER - JPC INTERFACE, Neely & Markov
AAA SUPER MODEM, Staff
COLOR USER NOTES, Nay
BASIC FORUM (CoCo), Upton
HELIX DM64 MEMORY SYS, Staff
WESTCHESTER Utilities, Ward
EXTENDED MEMORY TEST (Code), Comer
SORTED DIRECTORY, Rohlf
CoCo BASIC ROM 1.2 Addressing, Rosen
INFIX - POSTFIX EXP EVALUATOR, Baker
COMPARE - 6809 - Z80, Shearer
MODIFYING OKIDATA 84 PRI TER, Pass
Bit Bucket - Help - Classifieds

July

FLEX USER NOTES, Anderson
RUMORS & SUCH, DMW
SWTPC Distributors Meeting, DMW
COLOR USER NOTES, Nay
BASIC FORUM, Upton
'C' USER NOTES, Commo
OS9 USER NOTES, Dibble
SMALL DBMS (Inventory,etc), Lund
OMEGASOFT Pascal (Review), Blank
GIMIX G-28 - SWTPC DC-4, Johnson
SPEED UP - Color Disk, McDonald
TERM, Russell
FILE FIND (OS9), Childress
Bit Bucket - Help - Classifieds

August

FLEX USER NOTES, Anderson
COLOR USER NOTES, Nay
'C' USER NOTES, Commo
OS9 USER NOTES, Dibble
MACROBUFFER...Review, Tucker
CSC.Inc.XASM..Macro Sets.Review, Ward
RECORD LEVE LOCKOUT..OS9, Seng
PISTON SIM. - CT-82--BASIC&ASM, Placenza
DMS Notes..Update, Adams
XMPR...PROM ADAPTOR...Review, Staff
Bit Bucket - Help - Classifieds

September

FLEX USER NOTES, Anderson
COLOR USER NOTES, Nay
OS9 USER NOTES, Dibble
STASM ASSEMBLER..Review, Anderson
SELECTIVE DIR.CMD, Gitelson
XADR.....Review, Staff
CONDUIT...OS9..Review, Pass
PURCHASE ORDER WRITER, Cummings
EXTENDED MEMORY FOR OLD SYS, Scudiere
MP-C to MP-T for SWTPC, Wright
WHIMSICAL...A Compiler..Review, Anderson
TRANSPORT WITH DELIGHT, Fowler
GSPL..A Compiler..Review, Anderson
A SINGLE BOARD 6809 COMPUTER, Gitelson & Sansaska Sys
Bit Bucket - Help - Classifieds

October

RUMORS & SUCH, DMW
FLEX USER NOTES, Anderson
OS9 USER NOTES, Dibble
OS9 2D ANNUAL SEMINAR, DMW
OS9 FILE SORT, Strunk
C USER NOTES, Commo
COMPARE BASICS', Pass
CALCULATE DEC.HEX.BIN.Etc, Giesfeldt
BASLIST, Deal
PARAMOD..OS9, Bergvall
UNLOAD...OS9, Childress
TSC FORTRAN TEST, Killebrew
TEXT DECIMALIZATION MACROS, Mays
Bit Bucket - Help - Classifieds

November

FLEX USER NOTES, Anderson
OS9 USER NOTES, Dibble
BT-1 GRAPHICS BOARD REVIEW, Hanon
NEW COPY UTILITY, Taylor & Puglia
META LAB DAC1220 REVIEW, Ferguson
TSC ASSEMBLER PRINT UTILITY, Cole
XDMS DATA MANAGEMENT SYSTEM, Staff
UPGRADING A PERCOM SYSTEM, Bollinger
NOT JUST ANOTHER DUMB ROUTINE, Rees-Thomas
"HOLD" REVISED, Korte
A SMALL C, STS
Bit Bucket - Help - Classifieds

December

FLEX USER NOTES, Anderson
OS9 USER NOTES, Dibble
OS-9 CIS COBOL..Review, Dibble
C USER NOTES, Commo
SELECTRIC PRINTER DRIVERS.6800, Leavey
HEAD CLEAN - KEY LOCK, Aulicino
PL/9 - A/BASIC..Compared, Pass
MP-C to MP-T Revisited, Calhoun
'BASIC' SEARCH, Ramsey
JOURNAL Revisited.FLEX..CoCo, Deal
JUST..Review, Feintuch
RELINK, Condon
Bit Bucket - Help - Classifieds

SWTPC - Southwest Technical Products
CC - Color Computer

Note: Many of the items publshed under the **BIT BUCKET** column were voted the most useful by many readers. These letters, hints and kinks, suggestions and even sometimes (not often) gripes are what has directed us in our selection of articles each month.

We have attempted to select subject matter in relation to the percentage of different readers interest. However, it seems that some subjects have more reader interest, than we have received submisions from you. If we are to publish articles on subjects you are interested in, then we need **your input.** Unlike most magazines, 68 MICRO JOURNAL is **YOUR MAGAZINE!!!** This means that you the reader determines what we publish.

We forfeit thousands of dollars of advertising revenue each year because we screen advertised products to insure that they perform **as advertised.** Some 'bad apples' have slipped through (not many!). However, having tested (we order under an 'assumed' name many items not advertised in 68 MICRO JOURNAL) products advertised in other magazines, we haved decided to continue our screening requirements. There are other products that are advertised elsewhere (a very few) that are indeed excellent products. Therefore, it should not be assumed that just because it is not advertised in 68 MICRO JOURNAL it is of questionable quality. I am referring of course to products pertaining to the type computers that 68 MICRO JOURNAL is all about.

So if we are to continue to serve you as we have for the past five years plus, then I need your input on this also. If you have purchased a real 'lemon' let me know. I can assure you we will keep it on file and use the information accordingly. Of all the different groups of computers (Radio Shack, Apple, Commadore, IBM (personal) etc.) we have **less complaints** with our advertisers, than any of the others. We sure do not claim perfection, but I can assure you that you can place more **faith** in advertising in 68 MICRO JOURNAL, than **any** other computer magazine! We have received hundreds of pounds of letters stating as much, from you the readers.

DMW - - -

DYNAMITE REVIEW

By: Peter Dibble
517 Goler House
Rochester, NY 14620

Review of Dynamite
Overview

Dynamite is a disassembler for the 6809/6800 sold by Computer Systems Center. The version I tested runs under OS-9, but there are other versions for FLEX and UniFLEX. Disassemblers are able to convert a file of executable object (machine) code into a program in assembly language. It is important to realize that Dynamite won't work on intermediate code, such as Basic09 packed files, and it won't always convert object files into the original language. Dynamite can convert an executable object module generated by any language into assembly language. Even if the program was written in a higher level language like Pascal or C, Dynamite will only produce assembler.

If you have reliable software and don't like to dig around in your system much, you have no need for Dynamite. Don't waste your money. If you would like to fix (modify) your software, or just want to understand it as only someone with the source code can, Dynamite, or some other disassembler, is valuable. I have disassembled many pages of code by hand. Those hours of work qualify me to say that disassembly is just the type of work which should be left to computers.

Some Details

Dynamite can be used to get a quick look at source that could have generated an object file. The command:

DYNAMITE filename a will disassemble the module in the file called filename and send its output, which looks like the the output of an assembler, to the terminal. The "a" option tells Dynamite to give the ascii equivalent of each printable character it encounters during the disassembly. This simple disassembly is enough in many cases. If the module is more complicated than is easy to understand without meaningful labels, the next step is to help Dynamite do a better job of decoding the module until its output is understandable.

Some Details

Dynamite doesn't distinguish between data and instructions while disassembling. This results in some very strange output as blocks of constants are disassembled. Even the name of the program pointed to in the module header is decoded into assembly language instructions. The "a" option makes it easy to find the data areas, and Dynamite can be told where they are either through its standard input or in its command file. Once Dynamite knows where the data areas are, it will stop disassembling them as instructions. Instead, it will label the entries in the data area, and disassemble them into constants (fcb, fcc, ...).

When Dynamite is run without any guidance, it invents names for everything it encounters that might have had a name in the original program. Addresses, offsets, and immediate data all are given names. Names for immediate data and offsets are useful. Names for offsets in PCR instructions are VERY useful because, although different references to a location will have different PCR offsets, Dynamite resolves them to the same name.

An assembly language program more than about a page long is hard to read unless it has meaningful names. Dynamite gives names that consist of a letter and a number. More meaningful names can be assigned by using a label file.

Dynamite can use two classes of files with label definitions in the form of equates. It always uses a "system name" file which contains the names used for each OS9 call. When the instruction:

```
OS9 I$Open
```

is decoded the "I$Open" comes from the system name file. The second file full of label definitions is the "label file." The label file's name has to be given in the Dynamite command line. Each line in the label file is of the form:

```
label EQU value class
```

for example:

```
Init EQU $24 L
```

Where Init is the label, $24 is the value and "L" is the class. Initially eight label classes are defined:

D	Direct references
L	PCR references
X	Extended references
$	Hex constant
&	Decimal constant
@	Decimal or Hex constant depending on magnitude
^	ASCII constant
!	System function name

These classes are sufficient for a simple disassembly, but I found myself defining additional classes very soon. A class is defined by putting some labels in the label file with that class. All the unused letters A..Z can be used as new classes. For example, when I disassemble modules from OS-9, I usually have to define labels for offsets in the System Direct Page, and the process descriptor. For the System Direct Page the D class is fine, but for the process descriptor I have to define a new class. I usually use P.

Dynamite will use its default classes of labels wherever they are appropriate unless it is given instructions to use another class of label. A good disassembler needs to be able to assign labels to values very specifically. Although 8 is the offset of the P$User in the process descriptor control block, it wouldn't generally be a good idea to assign the name P$User to the value 8 throughout a program. Dynamite gives you two ways to limit the scope in which a label is used. A class of label is activated by a command of the form:

```
     <mode> <class> [<offset>] <range>
```

The mode is one of:

```
#1 - one byte immediate (any register)
#D - Immediate with Accumulator D
#X,#Y,#U,#S - Immediate with other registers
X,Y,U,S - Indexed by X,Y,U, or SP
D - Direct page
E - Extended addressing
R - Relative
```

The class is a default class, or one defined in the label file. The offset is added to a value before the proper label is looked up, then included in the disassembly listing. This would be used to generate instructions like:

```
lda #CR+$80
```

in the disassembly. The range gives the range of offsets from the start of the module being disassembled over which the mapping given by this command is in effect.

Commands can come either from standard input after Dynamite is started, or from a command file.

If the reason for disassembling a module is to learn how it works, the listing generated by Dynamite should be enough. If the goal is to revise the original program, Dynamite can generate a file which contains source which can be assembled with the Microware standard assembler, or any compatible assembler to give a module identical to the original.

The OS-9 version of Dynamite expects to disassemble 6809 instructions from a file with modules in OS-9 format, but there is an option which causes it to disassemble a file into 6800 instructions and another option which tells it to expect to find the module in Motorola or FLEX format instead of the usual OS-9 format.

Operation

I use Dynamite to sort of chew away at the edges of a program until I have it reduced to an understandable listing. First I let Dynamite have its head, and produce a listing using all its defaults. Using this listing, I start building the labels and commands files. At first I just define the data areas and a few labels. Then I go through a cycle of running Dynamite then using the output to refine and extend the contents of the commands and labels files until the listing satisfies me. Then I ask Dynamite to generate a file with the source in it. This file is the best I can do with Dynamite. It isn't well formatted, and has no comments. The final polishing has to be done with an editor.

Please realize that if you disassemble proprietary software (such as Dynamite itself) the same laws and moral obligations that should prevent you from passing out copies of the original program apply to the disassembled program.

Limitations

When I first tried to use Dynamite, I had a terrible time. I blamed the documentation. Determined not to be unfair, I sat down and read the manual from start to finish. I won't say it was easy reading, but once I had chewed my way through it I understood how to use DYnamite. The manual is a little brief for the manual of a program that does such tricky work, but it is complete. It is not set up to be skipped through!

Dynamite's advertising might lead a person to believe that disassembling a module with Dynamite is easy. You run Dynamite against a file and it falls apart into neat code. This is not true at all... disassembling a module is hard. You have to figure out all the tricks the person who wrote the program used. This is not too hard to do for a short, simple program, but long tangled modules are much harder to disassemble than they are to read in commented source form, and some modules are hard to understand even when the original source is in front of you.

It seems a little silly to design a disassembler with the ability to insert comments in its output, but Dynamite is such a complete product that I am a little disappointed that there is no way to include a "comment file" in the input for Dynamite. I understand that Computer Systems Center is working on this shortcoming.

Summary

I am very impressed with Dynamite. It does about as good a job of helping a person to disassemble a module as it can do. For example, if Dynamite finds that a label falls in the middle of an instruction, it throws in an ORG to adjust the PC so the label falls at the start of an instruction. This keeps data areas from throwing the disassembly out of whack; usually if there is a data area in a program, there is a reference to the first instruction after the data area which Dynamite can use to get itself lined up again if it hasn't been told that the data area is there and has gotten itself wrapped around the axle by trying to turn data into instructions.

Dynamite is designed to be useful for several different types of disassembly. The quick disassembly can be done without building any files. The most important information can be supplied interactively. Used this way Dynamite can produce a usable listing in just a few minutes. The full power and flexibility of the program shows up when a higher quality listing is the goal. Dynamite lends itself to the process of successive refinements that leads to a clear disassembly.

I don't recommend Dynamite for every OS-9 user. In fact, I imagine there are not many OS-9 users who have a need for this type of software, but for those who need a disassembler, Dynamite is everything it should be.

PACK

Joe Gottschall
1712 Leisure Lane
Glen Burnie, MD 21061
(301) 761-1151

Enclosed is a utility program called "PACK". It is a useful utility that will move all free sectors on a disk to one contiguous lump at the end of the disk. It will keep a file from becoming segmented if it is used BEFORE the file reaches the end of the disk. I suppose you are familiar with the phenomenon of a file creeping toward the end of the disk after many EDIT's, and then becoming scattered. The program was written with 5 1/4" disks in mind, but a friend of mine uses it regularly on his 8" drives under FLEX9. However, 8" disks take much longer to re-format, and a time savings may or may not be realized, depending on how many files are on the disk. A 5 1/4" disk can be packed in as little as 1'30", on my 6800 system running at 1.25 MHz.

The program supports single or double density, 5 1/4" or 8" drives. Two equates must be changed to convert to FLEX9.

I have been using the program for almost a year, and I have never had any problems with it. I hereby release the program "PACK" for publishing. I assume no responsibility whatsoever for it's use. Feel free to put the program on your Bulletin-Board system. I included a copy called "PACK-NO.SPC", just for this purpose. It has had the spaces removed by "REMSPC".

```
        OPT     PAG
        TTL     PACK
        PAS
*****************************************
*
* Written by: J. M. Sottschall
*             1712 Leisure Lane
*             Slen Burnie, MD 21061
*             (301) 761-1151
*
* Last edited: 11/14/83
*
*****************************************
*
* Purpose:
*
* This program will keep a disk from
* becoming too segmented by repeated
* EDIT's or SAVE's. It will move all
* FREE SPACE to the end of the disk.
* PACK must be used BEFORE the files
* become segmented.
*
*****************************************
*********** WARNING *******************
*
* Do NOT attempt to pack a disk
* which contains any non-contiguous
* files or directory sectors.
* If the disk contains more than 60
* files, you should use a disk doctor
* to see if the directory is non-
* contiguous.
*
******** ADDITIONAL WARNING! **********
*
* RANDOM files CANNOT be moved or they
* will be DESTROYED. Make sure any
* random files are on the front of
* the disk.
*
*****************************************
*
* SYNTAX:
*     +++PACK     Pack work drive
*     +++PACK 0   Pack specified drive
*
*****************************************
*
*   SYSTEM EQUATES
*

*FLEX    EQU    $C000     FOR FLEX9
FLEX     EQU    $A000     FOR FLEX2
*RESTOR  EQU    $D409     FOR FLEX9
RESTOR   EQU    $BE89     FOR FLEX2

FMS      EQU    FLEX+$1406     FILE MANASEMENT SYSTEM
FCB      EQU    FLEX+$0840     FILE CONTROL BLOCK
RPTERR   EQU    FLEX+$0D3F     REPEAT ERROR
WARMS    EQU    FLEX+$0D03     WARM START
WASN     EQU    FLEX+$0C0C     WORK DRIVE NUMBER
PSTRNG   EQU    FLEX+$0D1E     PRINT STRING
OUTADR   EQU    FLEX+$0D45     PRINT ADDRESS
PCRLF    EQU    FLEX+$0D24     PRINT CRLF
PUTCHR   EQU    FLEX+$0D16     WRITE CHARACTER
SETCHR   EQU    FLEX+$0D15     SET CHARACTER
NXTCH    EQU    FLEX+$0D27     NEXT LINE BUFFER CHARACTER
CURCHR   EQU    FLEX+$0C18     CURRENT LINE BUF. CHAR.
EOLCHR   EQU    FLEX+$0C02     END OF LINE CHAR.
ADDBX    EQU    FLEX+$0D36     ADD B RES. TO INDEX RES.
TTYSPS   EQU    FLEX+$0C09     TTYSET PAUSE CONTROL
MEMORY   EQU    FLEX+$0C28     FLEX MEMEND
VERIFY   EQU    FLEX+$1435     VERIFY FLAG

         ORG    FLEX+$0100
LFS      BRA    START
VERSN    FCB    3        VERSION

LOOP     FCB    0        LOOP INDICATOR
FILNUM   FCB    0        FILE NUMBER
TOTFIL   FCB    0        TOTAL FILES
LOWFS    FDB    0        LOW FREE SECTOR
LOWFA    FDB    0        LOW FILE ADDRESS
OLDSA    FDB    0        OLD START ADDRESS
NEWSA    FDB    0        NEW START ADDRESS
NEWTA    FDB    0        NEW STOP ADDRESS
XTEMP    FDB    0        TEMPORARY X REG.
LASTS    FDB    0        LAST TRACK-SECTOR ON DISK
SECOVF   FCB    0        SECTOR OVERFLOW INDICATOR
TTYPS    FCB    0        PAUSE STORAGE
MEMPT    FDB    0        MEMORY POINTER
MEMEND   FDB    0        MEMORY LIMIT
FCBPT    FDB    0        FCB POINTER
```

```
START   LDAA    TTYSPS
        STAA    TTYPS       STORE CURRENT PAUSE
        LDAA    #0          TURN PAUSE OFF
        STAA    TTYSPS
        LDX     MEMORY      GET FLEX MEMEND
        STX     MEMEND      STORE IT
        DEC     MEMEND      ADJUST FOR SAFETY
        LDAA    CURCHR      CHECK TO SEE IF
ST6     CMPA    #$0D        MORE CHARACTERS ARE
        BEQ     ST2         IN THE LINE BUFFER
        CMPA    EOLCHR      EOL OR $0D MEANS NO
        BNE     ST3         GET DRIVE NUMBER
ST2     LDAA    WASN        WORK DRIVE DEFAULT
ST5     STAA    FCB+3       STORE DRIVE #
        BRA     ST9
ST3     SUBA    #$30        STRIP ASCII
        CMPA    #0          VALID INPUTS ARE
        BLT     ST4         0,1,2 OR 3
        CMPA    #3
        BLE     ST5
ST4     JSR     NXTCH       KEEP SOINS TILL EOL
        BRA     ST6
ST9     JSR     PCRLF
        LDX     #DATA3      MAKE SURE DRIVE
        JSR     PSTRNS      IS CORRECT
        LDAA    FCB+3       SET #
        ADDA    #$30        MAKE ASCII FOR DISPLAY
        JSR     PUTCHR
        LDAA    #'?
        JSR     PUTCHR
        JSR     SETCHR      SET RESPONSE
        CMPA    #'Y         MUST BE A 'Y'
        BEQ     ST8
ST7     JMP     EXIT2       OR RESET TTYSET AND STOP
ST8     LDX     #FCB
        LDAA    #16         OPEN SYSTEM INFO. RECORD
        STAA    0,X
        JSR     FMS
        BEQ     CONT        CONTINUE-NO ERRORS
REPORT  JSR     RPTERR      REPEAT ERRORS NORMALLY
        BRA     ST7
CONT    LDX     #FCB
        LDAA    #7          SET INFO FROM FCB
        STAA    0,X
        JSR     FMS
        BNE     REPORT
        LDX     FCB+17      FIRST FREE SECTOR
        STX     FCB+30      MAKE CURRENT SECTOR
        STX     LOWFS       SET UP LOW FREE SECTOR
        LDX     FCB+26      LAST TRACK-SECTOR
        STX     LASTS       STORE FOR REFERENCE
*
* SECOVF REFLECTS THE NUMBER OF SECTORS
* ON ANY TRACK PLUS ONE. THIS IS USED
* FOR COMPARISON PURPOSES IN TSINC
*
        LDAA    LASTS+1
        INCA
        STAA    SECOVF      ONE PAST HIGH SECTOR
SSRD    LDX     #FCB
```

```
        LDAA    #9          SINSLE SECTOR READ
        STAA    0,X
        JSR     FMS
        BEQ     COMPAR
        BRA     REPORT
COMPAR  LDAA    FCB+64      IS THIS THE LAST ONE?
        BNE     CONT2       TRACK=0 IF IT IS
        BRA     DONE
CONT2   LDX     #LOWFS      COMPARE TO FIND THE
        LDAA    0,X         LOWEST FREE SECTOR
        CMPA    FCB+64      IF LOWFS IS LONER THAN
        BLO     NEXTLS      CURRENT LINK INDICATOR,
        BHI     NEWLS       KEEP SOING.
        LDAA    1,X         TRACK IS EQUAL-CHECK
        CMPA    FCB+65      SECTOR, TOO.
        BHI     NEWLS
NEXTLS  LDX     FCB+64      NEXT SECTOR IN FREE CHAIN
        STX     FCB+30
        BRA     SSRD
NEWLS   LDX     FCB+64
        STX     LOWFS       SAVE NEW LON FREE SECTOR
        BRA     NEXTLS      KEEP GOING TILL EOF
DONE    LDX     #DATA
        JSR     PSTRNS      OUTPUT LOWFS FOR REFERENCE
        LDX     #LOWFS
        JSR     OUTADR
*
* LFA - LOWEST FILE ADDRESS
* THIS SECTION SEARCHES THE DIRECTORY FOR THE
* FIRST FILE WHICH IS CLOSEST TO LOWFS.
* ANY FILES WHICH RESIDE AT A LOWER ADDRESS
* THAN LOWFS ARE COUNTED, BUT SKIPPED FOR
* COMPARISON. THIS ROUTINE IS LOOPED
* UNTIL ALL FILES HAVE BEEN MOVED
* TO AN ADDRESS LOWER THAN LOWFS.

LFA     CLR     LOOP        FIND LOWEST FILE ADDRESS
        CLR     FILNUM      (FIRST FILE PAST LOWFS)
        CLR     TOTFIL
        LDX     #FCB
        LDAA    #6          OPEN DIRECTORY
        STAA    0,X
        JSR     FMS
        BEQ     SRCH
        BRA     ERR1
SRCH    LDX     #FCB
        LDAA    #7          SET INFO RECORD
        STAA    0,X
        JSR     FMS
        BEQ     SRCH0
        BRA     ERR1
SRCH0   TST     FCB+4       FILENAME PRESENT?
        BNE     SRCH4       YES, SOMETHINS IS THERE
        BRA     SRCH
SRCH4   INC     TOTFIL      COUNT THE ENTRIES
        TST     FCB+4
        BMI     SRCH        FILE DELETED?
SRCH1   TST     LOOP        HAVE WE BEEN HERE BEFORE?
        BNE     SRCH2       THEN SKIP THIS PART
        LDAA    LOWFS
```

```
         CMPA   FCB+I7    ONLY FILES PAST LOWFS
         BLO    SRCH3     ARE VALID
         BHI    SRCH
SRCH5    LDAA   LOWFS+I   COMPARE SECTOR, TOO
         CMPA   FCB+IB
         BHS    SRCH      LOWFS IS HIGHER OR SAME
SRCH3    LDX    FCB+I7    FILE BEGIN ADDRESS
         STX    LOWFA     FIRST VALID DIRECTORY ENTRY
         INC    LOOP      SEARCH REST OF DIRECTORY
         LDAA   TOTFIL
         STAA   FILNUM    FIRST FILE TO BE MOVED
         BRA    SRCH      SCAN FOR MORE ENTRIES
SRCH2    LDAA   LOWFA     COMPARE FOR LOWEST
         CMPA   FCB+17    FILE ADDRESS
         BHI    NEWFA     IS LOWFA HIGHER?
         BLO    SRCH
         LDAA   LOWFA+I   DO SECTOR, TOO
         CMPA   FCB+IB
         BHI    NEWFA     GOT A NEW ONE!
         BRA    SRCH      KEEP GOING
NEWFA    LDAA   LOWFS     MUST BE HIGHER LOGICALLY
         CMPA   FCB+I7    THAN LOWFS
         BLO    NEW1
         BHI    SRCH      LOWFS IS HIGHER-DONT DO IT
         LDAA   LOWFS+I   DO SECTOR, TOO
         CMPA   FCB+IB
         BHS    SRCH
NEW1     LDX    FCB+I7    GET FILE BEGIN ADDRESS
         STX    LOWFA     SAVE NEW LOWFA
         LDAA   TOTFIL    KEEP TRACK OF
         STAA   FILNUM    WHICH ONE IS LOWFA!
         BRA    SRCH
ERR1     LDAA   I,X       ERROR TRAP FOR END OF
         CMPA   #B        FILE CONDITION IN DIRECTORY
         BEQ    DONE2     IF NOT AN 'B' THEN
         JMP    REPORT    REPORT OTHER ERROR
DONE2    TST    FILNUM    ANY FILES TO BE MOVED?
         BEQ    PACK      PACK WILL CHECK, TOO
         LDX    #DATA2
         JSR    PSTRNG
         LDX    #LOWFA    REPORT LOWFA
         JSR    OUTADR
         LDAA   #32
         JSR    PUTCHR
FILNAM   LDX    #FCB      PRINT FILENAME OF FILE
         LDAA   #6        TO BE MOVED
         STAA   0,X
         JSR    FMS
         BEQ    FIL2
         JMP    REPORT
FIL2     LDAB   FILNUM    DIRECTORY ENTRY NUMBER
         BEQ    PACK
FIL9     LDAA   #7        GET INFO RECORD
         STAA   0,X       FOR THAT ENTRY
         JSR    FMS
         BNE    ERROR
         DECB
         BNE    FIL9
FIL3     LDX    #FCB+4    START OF FILENAME
         LDAB   #B
```

```
         BSR    PRNAM
         LDAA   #'.
         JSR    PUTCHR
         LDX    #FCB+I2   EXTENSION
         LDAB   #3
         BSR    PRNAM
PACK     BRA    PACK1     SKIP SUBROUTINE
ERROR    JMP    REPORT    HERE FOR CONVENIENCE
*
* PRNAM PRINTS THE NUMBER OF LETTERS
* INDICATED IN THE B REG. AND POINTED TO
* BY THE X REG.
*
PRNAM    LDAA   0,X
         BNE    PRNAM3
         LDAA   #32       SPACE IF BYTE IS ZERO
PRNAM3   JSR    PUTCHR
         INX
         DECB
         BNE    PRNAM
         RTS
*
* LOWFA AND LOWFS ARE KNOWN.
* THE FILE IS READ INTO
* USER MEMORY. IF AN ERROR OCCURS
* DURING THE MOVE, THE FILE COULD
* PROBABLY STILL BE RECOVERED, BUT ONLY IF
* IT WAS NOT WRITTEN ON TOP OF DURING
* THE PACK PROCESS. IF ANY ERRORS OCCUR,
* I RECOMMEND YOU USE 'COPY' TO SALVAGE
* THE DISK.

PACK1    LDAB   FILNUM    MAKE SURE FILE IS TO BE
         BNE    PACK2     PACKED, OTHERWISE
         JMP    FINISH    FINISH UP.
PACK2 LDX #0
         STX    MEMPT     CLEAR POINTER
         LDX    LOWFS     GET LOW FREE SECTOR
         STX    NEWSA     NEW START ADDR.
         STX    NEWTA     NEW STOP ADDR.
         LDAA   #$FF
         LDX    #FCB
         STAA   59,X      NO SPACE COMPRESSION
         LDX    17,X      STARTING DISK ADDRESS
PACK5    STX    FCB+30    PREP TO READ
         LDX    #FCB
         LDAA   #9        SINGLE SECTOR READ
         STAA   0,X
         JSR    FMS
         BNE    ERROR
* STORE THE READ-IN SECTOR IN USER MEMORY
PACK3    LDX    #FCB+64
PACK4    STX    FCBPT     PREP FCB POINTER
         LDAA   0,X       GET BYTE FROM SECTOR
         LDX    MEMPT     LOAD MEMORY POINTER
         STAA   0,X       SAVE BYTE
         INX
         STX    MEMPT     NEXT MEMORY LOCATION
         LDX    FCBPT
         INX
```

```
        CPX     #FCB+320  END OF SECTOR?
        BNE     PACK4
*CHECK FOR END-OF-FILE OR BUFFER FULL
        LDX     FCB+64    IF END-OF-FILE,
        BEQ     WRTFIL    WRITE IT OUT
        LDX     MEMPT     GET POINTER
        CPX     MEMEND    PAST MEMEND?
        BGE     PACK6     MEMPT-MEMEND >=0?
        LDX     FCB+64    GET NEXT SECTOR LINK
        BRA     PACK5
* FILE WON'T FIT IN MEMORY -
* SAVE THE ADDRESS OF THE
* NEXT SECTOR IN THE FILE
PACK6   LDX     FCB+64    NEXT SECTOR LINK
        STX     OLDSA     SAVE IT

WRTFIL  LDX     #FCB+64
        STX     FCBPT     PREP FCB POINTER
        LDX     #0
        STX     MEMPT     CLEAR MEMORY POINTER
WRT2    LDAA    0,X       GET BYTE
        INX
        STX     MEMPT     BUMP POINTER
        LDX     FCBPT
        STAA    0,X       SAVE BYTE IN FCB
        INX
        STX     FCBPT     BUMP POINTER
        CPX     #FCB+320  END OF SECTOR?
        BEQ     WRT3
        LDX     MEMPT
        BRA     WRT2
WRT3    LDX     LOWFS
        STX     FCB+30    PREP TO WRITE
        JSR     TSINC     TRACK,SECTOR INCREMENT
        STX     LOWFS     KEEP THIS CURRENT!
        TST     FCB+64    EOF?
        BEQ     CLEAN     CLEAN IT UP
        STX     FCB+64    LINK SECTOR
        STX     NEWTA     NEW STOP ADDR.
        JSR     WRSS      WRITE SINGLE SECTOR
        LDX     #FCB+64
        STX     FCBPT     PREP FCB POINTER
        LDX     MEMPT
        CPX     MEMEND    PAST MEMEND?
        BLE     WRT2      NO, KEEP GOING
* SET POINTERS FOR THE NEXT PASS
* ON A LARGE FILE
        LDX     #0
        STX     MEMPT
        LDX     OLDSA     CONTINUATION POINT
        JMP     PACK5
CLEAN   JSR     WRSS      WRITE FINAL SECTOR
        LDX     #FCB
        LDAA    #6        OPEN DIRECTORY
        STAA    0,X
        JSR     FMS
        BEQ     CL1
        JMP     REPORT
CL1     LDAB    FILNUM    GET APPROPRIATE
CL2     LDAA    #7        INFO RECORD
```

```
        STAA    0,X
        JSR     FMS
        BEQ     CL3
        JMP     REPORT
CL3     DECB              GO TILL B=0
        BNE     CL2
        LDX     #FCB
        LDAB    49,X      DATA OFFSET BYTE
        LDX     #FCB+77   POINT TO FIRST FILE ADDRESS
        JSR     ADDBX     ADD DATA OFFSET
        LDAA    NEWSA
        STAA    0,X       SECTOR BUFFER MUST BE
        LDAA    NEWSA+1   CHANGED DIRECTLY. THE
        STAA    1,X       PUT INFO. RECORD FUNCTION
        LDAA    NEWTA     WILL NOT CHANGE THESE
        STAA    2,X       BYTES
        LDAA    NEWTA+1
        STAA    3,X
        LDX     NEWSA
        STX     FCB+17    JUST IN CASE, CHANGE
        LDX     NEWTA     THESE, TOO
        STX     FCB+19
        LDX     #FCB
        LDAA    #B        PUT INFO REC. BACK
        STAA    0,X
        JSR     FMS
        BEQ     CL4
        JMP     TOBAD2    DIRECTORY PROB. DAMAGED
CL4     JMP     LFA       FIND LOWEST FILE AGAIN
FINISH  LDX     #FCB
        LDAA    #16       OPEN SYSTEM RECORD
        STAA    0,X
        JSR     FMS
        BEQ     FI1
        JMP     REPORT
FI1     LDAA    #7        SET INFO REC
        STAA    0,X
        JSR     FMS
        BEQ     FI2
        JMP     REPORT
FI2     LDX     FCB+26    LAST TRACK,SECTOR
        STX     LASTS
        LDX     #FCB      LOCATE THE TRACK-SECTOR
        LDAB    49,X      INFO. IN THE SECTOR
        LDX     #FCB+77   BUFFER ITSELF.
        JSR     ADDBX     ADD DATA OFFSET
        LDAA    LOWFS
        STAA    0,X       MODIFY SECTOR BUFFER
        LDAA    LOWFS+1   DIRECTLY
        STAA    1,X
        LDAA    LASTS
        STAA    2,X
        LDAA    LASTS+1
        STAA    3,X
        LDX     LOWFS
        STX     FCB+17    JUST IN CASE
        LDX     LASTS
        STX     FCB+19
        LDX     #FCB
        LDAA    #8        PUT INFO REC. BACK
        STAA    0,X
        JSR     FMS
        BEQ     FI3
        BRA     TOBAD3
FI3     LDX     #RDATA      RE-FORMAT REST OF DISK
        JSR     PSTRNG      THIS TAKES AWHILE.
        CLR     VERIFY      SPEEDS THINGS UP
        LDX     #FCB+64
        LDAA    #0
FI4     STAA    0,X         CLEAR OUT BUFFER
        INX
        CPX     #FCB+320
        BNE     FI4
FI5     LDX     LOWFS
        STX     FCB+30      PREP TO WRITE
        CPX     LASTS       LAST SECTOR?
        BEQ     EXIT
        LDX     LOWFS
        BSR     TSINC       PREP FOR LINKAGE
        STX     LOWFS
        STX     FCB+64      LINK TO NEXT
        LDX     #FCB
        LDAA    #10         SINGLE SECTOR WRITE
        STAA    0,X
        JSR     FMS
        BEQ     FI5
        BRA     TOBAD4      CLEAN-UP ERROR
EXIT    CLR     FCB+64      SET EOF INDICATION
        CLR     FCB+65
        COM     VERIFY      TURN IT BACK ON
        LDX     #FCB
        LDAA    #10         S.S. WRITE
        STAA    0,X
        JSR     FMS
        BNE     TOBAD5      ERROR ON LAST SECTOR
        JSR     RESTOR      SEEK TO TRACK 00
        LDX     #DATA4      ALL DONE!
        JSR     PSTRNG
EXIT2   LDAA    TTYPS       RESTORE TTYSET PAUSE
        STAA    TTYSPS
        JMP     WARMS
TOBAD5  LDX     #DATA5
TOOBAD  JSR     PSTRNG
        JMP     REPORT      REPORT ERROR THRU FLEX
TOBAD4  LDX     #DATA6
        BRA     TOOBAD
TOBAD3  LDX     #DATA7
        BRA     TOOBAD
TOBAD2  LDX     #DATA8
        BRA     TOOBAD
TOBAD   LDX     #DATA9
        BRA     TOOBAD
*
* TSINC WILL INCREMENT THE TRACK AND
* SECTOR ADDRESS WHILE CONFORMING TO
* THE NUMBER OF SECTORS PER TRACK.
* THE ADDRESS IS STORED IN THE X
* REGISTER ON ENTRY AND RETURNED
* IN THE X REG. ON EXIT.
*
TSINC   INX                 TRACK-SECTOR INCREMENT
        STX     XTEMP
        LDAA    XTEMP+1
        CMPA    SECOVF      OVER LAST SECTOR?
        BNE     TSRET
        LDAA    #1          FIRST SECTOR
        STAA    XTEMP+1     OF
        INC     XTEMP       NEXT TRACK
        LDX     XTEMP
TSRET   RTS
WRSS    LDX     #FCB        WRITE SINGLE SECTOR
        LDAA    #10
        STAA    0,X
        JSR     FMS
        BEQ     WRSI
        PULA                FIX STACK
        PULA
        BRA     TOBAD       FILE PROBABLY DAMAGED
WRSI    RTS

DATA    FCC     'Lowest free sector '
        FCB     4
DATA2   FCC     'Moving from '
        FCB     4
DATA3   FCC     'REMEMBER: RANDOM files CANNOT be moved!'
        FDB     $0D0A
        FCC     'Pack drive 0'
        FCB     4
DATA4   FCC     'Pack complete'
        FCB     4
DATA5   FCC     'ERROR while writing very last sector.'
        FDB     $0D0A
        FCC     'Use "COPY" to salvage the disk.'
        FCB     4
DATA6   FCC     'An ERROR occured while reformatting.'
        FDB     $0D0A
        FCC     'Use "COPY" to salvage the disk.'
        FCB     4
DATA7   FCC     'System Info Record damaged'
        FCB     4
DATA8   FCC     'Directory probably damaged'
        FCB     4
DATA9   FCC     'ERROR during Single-Sector-Write.'
        FCC     'File probably damaged'
        FCB     4
RDATA   FCC     'All files moved.'
        FDB     $0D0A
        FCC     'Re-formatting. Please be patient.'
        FCB     4
        END     LF6
```

WORD.BAS & XON .OVL

Chris Robinson

A couple of progs that may be of use to someone.The 'words.bas' prog is yet another bigprint prog,but i find it more usefull.The 'xon.ovl' prog allows the normal 'flex' escape routine to operate but adds 'xon/xoff'capability.How about more assembler routines i.e Ron Andersons never to be aired maths routines.Keep up the good work on an excellant magazine.

```
1 REM
2 REM This BIGPRINT program was developed and written
3 REM by Chris Robinson
4 REM P.O box 9276
5 REM Hamilton
6 REM New Zealand
7 REM
8 REM

-------------------------------------------------------------
50 REM characters are displayed in a 7X11 grid with lower
case descenders.
51 REM each value in the data statemnets contains the ^2
value of each row.
52 REM 65=bit 2^7 and 2^1 are on i.e   X00000X
53 REM 18=bit 2^5 and 2^2 are on i.e   00X00X0
54 REM 127=2^7+2^6+2^5+2^4+2^3+2^2+2^1 XXXXXXX
98 REM

-------------------------------------------------------------
99 REM system equates
100 CL$=CHR$(HEX("1A"))                      :REM clear screen
character
110 ES$=CHR$(HEX("1B"))                      :REM escape
character
120 GR$=ES$+"$"                              :REM put terminal
into graphics mode
130 NG$=ES$+"%"                              :REM out of
graphics mode
140 BL$=GR$+CHR$(HEX("7F"))+NG$              :REM block
character
498 REM

-------------------------------------------------------------
499 REM start prog.set up array's,fill,get parameters.
500 PRINT CL$                                :REM clear screen
1000 DIM B%(95,11):DIM B$(95):DIM A%(11)     :REM alocate
storage
1010 GOSUB 5400                              :REM fill quick
reference array
1020 INPUT "Width magnification factor (1 to 30)   ";W$:IF
VAL(W$)<1 OR VAL(W$)>30 THEN 1020 ELSE W%=VAL(W$)
1040 INPUT "Height magnification factor (I to 30)  ";H$:IF
VAL(H$)<1 OR VAL(H$)>30 THEN 1040 ELSE H%=VAL(H$)
1060 INPUT "Use 't'he character,'b'lock or 'u'ser supplied
(t,b or u)  ";C$
1080 IF C$="T" OR C$="t" OR C$="B" OR C$="b" OR C$="U" OR
C$="u" THEN 1090 ELSE 1060
1090 IF C$="T" OR C$="t" THEN C$="-2"
1095 IF C$="B" OR C$="b" THEN C$="-1"
1100 IF C$="U" OR C$="u" THEN INPUT"Which character would
you like to use ";C$
1110 INPUT "Width of page (40 to 160)  ";P%:REM arbitary
values
1120 IF P%<40 OR P%>160 THEN 1110
1130 CLOSE 0
1140 INPUT "Print 'a'cross the page or 'd'own the page (a or
d)  ";O$
1150 IF O$="A" OR O$="a" OR O$="D" OR O$="d" THEN 1160 ELSE
1140
1160 INPUT "Output to the 's'creen or to the 'p'rinter or
'q'uit (s,p or q)  ";P$
1170 IF P$="Q" OR P$="q" THEN 10000
1180 IF F$="S" OR P$="s" OR P$="P" OR P$="p" THEN 1190 ELSE
1160
1190 IF F$="P" OR P$="p" THEN 2000 ELSE 2040
2000 OPEN "S.CMD" AS 0                       :REM open printer
file
2005 IF C$="-1" THEN C$="-2"                 :REM force 'the
character' if for printer
2010 INPUT "Is the paper aligned (y/n)  ";X1$
2020 IF X1$="Y" OR X1$="y" THEN 2040 ELSE 2010
2040 INPUT "Do you want to continue (y/n)  ";X1$
2050 IF X1$="Y" OR X1$="y" THEN 2100
2060 IF X1$="N" OR X1$="n" THEN 10000 ELSE 2040
2100 PRINT CL$
2110 T$=""
2160 PRINT "Please type in the text you wish to display "
2180 INPUT LINE T$
2200 IF O$="D" OR O$="d" THEN 3010
2210 S%=((LEN(T$)*7)*W%)+((LEN(T$)-1)*(1*W%)):REM find
centre
2220 IF S%<=P% THEN 2450
2230 PRINT "The length of text times the magnification
factor is to wide for the page !!"
2240 PRINT "Please re-enter. ":GOTO 2110
2450 PRINT CL$
2500 WI%=P%-S%
2510 IF W1%=0 THEN SI%=0 ELSE S1%=INT(W1%/2)
2550 FOR I%=1 TO 11                          :REM depth of
character
2560   FOR M%=1 TO H%                        :REM height amp
factor
2570     PRINT#0,TAB(S1%);"";                :REM centre
justify
2580     FOR K%=1 TO LEN(T$)                 :REM no of chars
2590       GOSUB 5100                        :REM get working
character
2595       C%=A%(I%)
2600       Z%=64
2610       FOR J%=1 TO 7
2620         C%=C%-Z%
2630         IF A%(0)=1 THEN D$=BL$ ELSE D$=CHR$(A%(0))
2634         IF C%<0 THEN D$=" "
2640         FOR L%=1 TO W%:PRINT#0,D$;
2650         NEXT L%
2660         IF C%<0 THEN C%=C%+Z%
2670         Z%=Z%/2
2680       NEXT J%
2685       IF K%=LEN(T$) THEN 2720
2690       FOR K2%=1 TO (1*W%)
2700         PRINT#0," ";
```

```
2710        NEXT K2%
2720      NEXT K%
2730      PRINT#0
2740    NEXT M%
2750 NEXT I%
2770 GOTO 2040
3000 REM
3010 IF 11*H%<=P% THEN 3050
3020 PRINT "The height of the characters times the
magnification factor "
3030 PRINT "is too wide for the page size ......... Please
re-enter !!!"
3040 GOTO 1020
3050 H1%=P%-(11*H%)
3060 IF H1%=0 THEN S1%=0 ELSE S1%=INT(H1%/2)
3070 FOR K%=1 TO LEN(T$)                    :REM do for each
char
3080   GOSUB 5100
3090   Z%=64
3100   FOR I%=1 TO 7                        :REM do for each
bit
3110     FOR J%=1 TO W%
3120       PRINT#0,TAB(S1%);"*";
3130       FOR L%=11 TO 1 STEP -1
3140         C%=A%(L%)
3150         C%=C%-Z%
3160         IF A%(0)=1 THEN D$=BL$ ELSE D$=CHR$(A%(0))
3170         IF C%<0 THEN D$=" "
3180         FOR M%=1 TO H%:PRINT#0,D$;
3190         NEXT M%
3200       NEXT L%
3210       IF J%<>W% THEN 3250
3220       FOR N%=1 TO 11
3230         IF A%(N%)>=Z% THEN A%(N%)=A%(N%)-Z%
3240       NEXT N%
3250       PRINT#0
3260     NEXT J%
3270     Z%=Z%/2
3280   NEXT I%
3290   IF K%=LEN(T$) THEN 3330
3300   FOR M%=1 TO (1*W%)
3310     PRINT#0
3320   NEXT M%
3330 NEXT K%
3350 GOTO 2040
4000 DATA " ",0,0,0,0,0,0,0,0,0,0,0
4010 DATA A,28,34,65,65,65,127,65,65,65,0,0
4020 DATA a,0,0,0,28,2,62,66,66,61,0,0
4030 DATA B,126,33,33,33,62,33,33,33,126,0,0
4040 DATA b,64,64,64,92,98,66,66,98,92,0,0
4050 DATA C,30,33,64,64,64,64,64,33,30,0,0
4060 DATA c,0,0,0,60,66,64,64,66,60,0,0
4070 DATA D,124,34,33,33,33,33,33,34,124,0,0
4080 DATA d,2,2,2,58,70,66,66,70,58,0,0
4090 DATA E,127,64,64,64,120,64,64,64,127,0,0
4100 DATA e,0,0,0,60,66,126,64,64,60,0,0
4110 DATA F,127,64,64,64,120,64,64,64,64,0,0
4120 DATA f,12,18,16,16,124,16,16,16,16,0,0
4130 DATA G,30,33,64,64,64,79,65,33,30,0,0
4140 DATA g,0,0,0,58,70,66,70,58,2,66,60
4150 DATA H,65,65,65,65,127,65,65,65,65,0,0
4160 DATA h,64,64,64,92,98,66,66,66,66,0,0
4170 DATA I,62,8,8,8,8,8,8,8,62,0,0
4180 DATA i,0,8,0,24,8,8,8,8,28,0,0
4190 DATA J,31,4,4,4,4,4,4,68,56,0,0
4200 DATA j,0,0,0,6,2,2,2,2,2,34,28
4210 DATA K,65,66,68,72,80,104,68,66,65,0,0
4220 DATA k,64,64,64,68,72,80,104,68,66,0,0
4230 DATA L,64,64,64,64,64,64,64,64,127,0,0
4240 DATA l,24,8,8,8,8,8,8,8,28,0,0
4250 DATA M,65,99,85,73,73,65,65,65,65,0,0
4260 DATA m,0,0,0,118,73,73,73,73,73,0,0
4270 DATA N,65,97,81,73,69,67,65,65,65,0,0
4280 DATA n,0,0,0,92,98,66,66,66,66,0,0
4290 DATA O,28,34,65,65,65,65,65,34,28,0,0
4300 DATA o,0,0,0,60,66,66,66,66,60,0,0
4310 DATA P,126,65,65,65,126,64,64,64,64,0,0
4320 DATA p,0,0,0,92,98,66,66,98,92,64,64
4330 DATA Q,28,34,65,65,65,73,69,34,29,0,0
4340 DATA q,0,0,0,58,70,66,66,70,58,2,2
4350 DATA R,126,65,65,65,126,72,68,66,65,0,0
4360 DATA r,0,0,0,92,98,64,64,64,64,0,0
4370 DATA S,62,65,64,64,62,1,1,65,62,0,0
4380 DATA s,0,0,0,60,66,48,12,66,60,0,0
4390 DATA T,127,8,8,8,8,8,8,8,8,0,0
4400 DATA t,0,16,16,124,16,16,16,18,12,0,0
4410 DATA U,65,65,65,65,65,65,65,65,62,0,0
4420 DATA u,0,0,0,66,66,66,66,70,58,0,0
4430 DATA V,65,65,65,34,34,20,20,8,8,0,0
4440 DATA v,0,0,0,65,65,65,34,20,8,0,0
4450 DATA W,65,65,65,65,73,73,85,99,65,0,0
4460 DATA w,0,0,0,65,73,73,73,73,54,0,0
4470 DATA X,65,65,34,20,8,20,34,65,65,0,0
4480 DATA x,0,0,0,66,36,24,24,36,66,0,0
4490 DATA Y,65,65,34,20,8,8,8,8,8,0,0
4500 DATA y,0,0,0,66,66,66,70,58,2,66,60
4510 DATA Z,127,1,2,4,8,16,32,64,127,0,0
4520 DATA z,0,0,0,126,4,8,16,32,126,0,0
4530 DATA "0",62,65,67,69,73,81,97,65,62,0,0
4540 DATA "1",8,24,40,8,8,8,8,8,62,0,0
4550 DATA "2",62,65,1,2,28,32,64,64,127,0,0
4560 DATA "3",62,65,1,1,30,1,1,65,62,0,0
4570 DATA "4",2,6,10,18,34,66,127,2,2,0,0
4580 DATA "5",127,64,64,124,2,1,1,66,60,0,0
4590 DATA "6",30,32,64,64,126,65,65,65,62,0,0
4600 DATA "7",127,65,2,4,8,16,16,16,16,0,0
4610 DATA "8",62,65,65,65,62,65,65,65,62,0,0
4620 DATA "9",62,65,65,65,63,1,1,2,60,0,0
4630 DATA !,8,8,8,8,8,0,0,8,8,0,0
4640 DATA @,30,33,77,85,85,94,64,32,30,0,0
4650 DATA #,20,20,20,127,20,127,20,20,20,0,0
4660 DATA $,8,63,72,72,62,9,9,126,8,0,0
4670 DATA %,32,81,34,4,8,16,34,69,2,0,0
4680 DATA ^,8,20,34,65,0,0,0,0,0,0,0
4690 DATA &,56,68,68,40,16,41,70,70,57,0,0
4700 DATA *,0,8,73,42,28,42,73,8,0,0,0
4710 DATA (,4,8,16,16,16,16,16,8,4,0,0
4720 DATA ),16,8,4,4,4,4,4,8,16,0,0
4730 DATA -,0,0,0,0,127,0,0,0,0,0,0
4740 DATA _,0,0,0,0,0,0,0,0,127,0,0
```

```
4750 DATA +,0,8,8,8,127,8,8,8,0,0,0
4760 DATA =,0,0,0,30,0,30,0,0,0,0,0
4770 DATA ~,48,73,6,0,0,0,0,0,0,0,0
4780 DATA `,24,24,8,4,0,0,0,0,0,0,0
4790 DATA !,8,8,8,0,0,8,8,8,0,0,0
4800 DATA \,0,64,32,16,8,4,2,1,0,0,0
4810 DATA ],60,4,4,4,4,4,4,4,60,0,0
4820 DATA [,60,32,32,32,32,32,32,32,60,0,0
4830 DATA :,0,0,0,24,24,0,0,24,24,0,0
4840 DATA ;,0,0,0,24,24,0,0,24,24,16,32
4850 DATA "'",12,12,8,16,0,0,0,0,0,0,0
4860 DATA '"',36,36,36,0,0,0,0,0,0,0,0
4870 DATA <,4,8,16,32,64,32,16,8,4,0,0
4880 DATA ",",0,0,0,0,0,0,0,24,24,16,32
4890 DATA >,16,8,4,2,1,2,4,8,16,0,0
4900 DATA ".",0,0,0,0,0,0,0,24,24,0,0
4910 DATA ?,30,33,33,1,6,8,8,0,8,0,0
4920 DATA /,0,1,2,4,8,16,32,64,0,0,0
4930 DATA },24,4,4,4,2,4,4,4,24,0,0
4940 DATA {,12,16,16,16,32,16,16,16,12,0,0
5000 REM
5100 REM transfer subroutine.find character
5110 X1$=MID$(T$,K%,I)
5120 FOR II%=I TO 96
5130   IF 11%=96 THEN 9990
5140   IF X1$=B$(I1%) THEN GOTO 5300
5150 NEXT I1%
5160 RETURN
5299 REM put character into working array.
5300 FOR K1%=I TO I1
5310   A%(KI%)=B%(11%,K1%)
5320 NEXT K1%
5330 IF C$="-1" THEN A%(0)=1
5340 IF C$="-2" THEN A%(0)=ASC(XI$)
5350 IF LEN(C$)<2 THEN A%(0)=ASC(C$)
5360 RETURN
5399 REM transfer data to quick reference array.
5400 PRINT "Hang on for a second while I relearn the
alphabet............."
5410 RESTORE
5420 FOR I%=1 TO 95
5425   PRINT ".";
5430   READ B$(I%)
5440   FOR K%=1 TO 11
5450     READ B%(I%,K%)
5460   NEXT K%
5470 NEXT I%
5480 PRINT:PRINT "Right here we go.......now.......":PRINT
5490 RETURN
9990 PRINT "One of the characters does not exist
!!!!!!!!!!!!"
10000 INPUT "Do you want to 'q'uit or to 'r'estart (q or r)
";X1$
10005 PRINT CL$
10010 IF X1$="R" OR X1$="r" THEN 1020
10020 IF XI$="Q" OR XI$="q" THEN 10030 ELSE 10000
10030 CLOSE 0
10040 END
```

```
*
*               new xon/xoff routine
*
* overlay called by `get xon.ovl' at startup.
*
esc_routine     equ     $ce90             flex escape routine
inchar          equ     $d381             flex get chracter routine
*
                org     $d37d             break into existing routine
                bra     newroutine
*
                org     $f000             put in DMA space
newroutine      sta     1,x               old routine (print character)
                bsr     esc_routine       use existing  escape routine
                cmpa    #$13              DC3 char (X off)
                bne     finish            if previous ttyesc or not xoff
getcharloop     bsr     inchar            was xoff now wait for xon
                cmpa    #$11              X on ???
                bne     getcharloop       no
                clra                      clra only if previous xoff/xon
finish          puls    pc,x,b            return
*
                end
```

Chris Robinson
C/-P.O Box 9276
Hamilton
New Zealand

UNIVERSAL SINGLE BOARD

The XL 68 S is a single board computer which was developed on the basis of the existing XL 68 Euroboard system comprising about 25 cards 10 x 16 cm with 64 pin/socket connectors each.

It was the extensive use of those high priced connectors, the repeated need of power consuming buffer chips and the waste of board space which triggered the idea.

The two guiding principles throughout the development were 1) the exploitation of Motorola's 6809's capabilities and 2) additional hardware enhancements for increased speed and programming power.

Although any 6809 operating system could be implemented there are at least three on the market which would fit directly with little jumper movement: FLEX or UNIFLEX from TSC or OS-9 available from Microware. As a matter of fact an external toggle switch gives you fast and easy choice between the two worlds assuming you got the monitor ROMs plugged in. With cross assemblers in the wings we can all hope for greater activities with both operating systems.

The dimensions of the XL 68 S board are 31 by 42 cm so the longest side would still fit into a 19" rack mount.

All peripheral connectors are of the printed circuit type and are soldered to the long edges of the board.

The board is divided into function areas which surround the CPU block located at the center. Also in the center is the bus connector for a twofold purpose. It allows for diagnostics while each function area can be disabled with jumpers and it serves as an expansion connector for future sandwiching a second board level onto the main board. At present we will only be concerned with the main board.

The CPU is Motorola's 6809E. The E-version was chosen for multiprocessing applications with additional 6809E's through the above mentioned bus connector.

Memory management which could be disabled deals with 4K and 64K blocks (exchanging buffers and moving jumpers allows for SWTP's DAT configuration assuming the use of their S-BUG). A DIL socket is provided for signature analysis etc. There is a watch dog circuit in this area of the board which, when enabled, will trigger a monostable at power-on. This monostable would have to be retriggered through software at strategic points of your program in order not to RESET the computer. This is some kind of alarm system often used with industrial applications to prevent the computer from "running wild".

Next to the CPU block is the main decoding area responsible for generating chip selects or area selects for the whole board as well as predecoded signals for future I/O and A/O and O/A converters. Needless to say that there are jumpers connected with it.

The PROM area contains sockets for 22K. Two of those sockets can be configured for a total of 4K CMOS RAM with on-board battery back-up to save data at power-fall. There are two sockets at F800 for switch-selecting operating systems.

The 6 DB-25 connectors on one side of the board directly feed to 5 terminals (or other serial applications) and one (parallel) printer. Through jumpers and an additional DC/DC converter, two of the serial ports can be charged to opto-isolated 20 mA interfaces.

The 256K dynamic RAM is physically located at 0 0000 to 0 7FFF, 1 0000 to 1 7FFF etc. in eight 32K blocks. One could plug in only 64K byte or one could disable it completely and use the above 4K CMOS RAM for test purposes.

The floppy disk controller for 8" drives features DMA for double density and makes use of the Western Digital controller set.

The 5 1/4" working mode is being provided but has yet to be tried out. An adapter plug/receptacle to convert from the standard 8" to the 5 1/4" connector would also be needed.

The next board edge connector is a 64 pin/socket type catering the three PIA's. Their A0/A1 lines could be used as is or interchanged for easy 2 byte operation.

An IEEE-488 GPIB interface serves as controller, listener or talker. To free the CPU from too much GPIB work and to speed up transfer by a factor of 4 a second DMA controller may be used in conjunction with the interface. The decision lies with the software. The connector is a DB-25.

Two of the remaining DMA channels of the same controller together with 1K byte of private RAM constitute a fast way of transferring large amounts of data with very little CPU time. The transfer rate is 500 byte/msec.

There is one DMA channel left. Its control lines and the data bus are accessible from the outside world through another DB-25 connector, opening up more data transfer possibilities. The same connector also carries an additional PIA byte with optional buffer and a single line outputting a 4 msec pulse when selecting a speciel address.

An OKI 5832 delivers date and time when so told via special program and PIA and it will keep time with its own on-board battery supply.

The most costly single item on the board is the 9511 or 9512 arithmetic processor and may seldom be used; but taking up little space, the socket and a few more components were incorporated to offer, if so desired, 10:1 improvement over software arithmetic.

Last not least there are two 6840 timer chips. Besides their many internal uses one of the timers has, if configured via jumpers, its input and output signals fed through some unused pins of a DB-25 connector otherwise used for RS-232.

As for expansion: I already mentioned the bus connector for multi processing. But I'm also thinking of high resolution graphics and/or winchester controller.

Furthermore, a 40 pin Molex is provided with the necessary control and data lines for A/D and D/A conversion.

For I/O expansion 4 DIL sockets with predecoded select lines would feed 5 ACIAs or 5 PIAs.

As pointed out earlier all those expansions would fit on a second board level realized in form of independent smaller boards.

With a power consumption of about 20 Watts the board would need no additional cooling when mounted vertically (i.e. behind the drawer housing of a type writer table. The housing itself could contain two floppies and the power supplies).

By now I expect that while reading all this you came up with a few applications of your own. Well, that's the idea!

Now for some figures: the board's real estate has been reduced by about 38 % compared to conventional set-ups. The same measures reduced the power by about 30 %. The expense for the motherboard with its plug/socket combination was eliminated and so were a number of redundant components like buffers, decoding circuitry etc. The resultant advantages, for the given number of functions, are considerable savings concerning material and power supply, increased reliability and higher life expectancy due to the lack of interconnecting mechanics and lower temperature.

With the higher complexity of the board more thought was given to diagnostics. Each function area can be disabled seperately with one jumper. The same goes for V_{cc}.

As for economics, the degree of assembly can be made to match the application.

A final word about the availability of the XL 68 S: I am looking for a computer or kit manufacturer who might be interested in some license agreement.

At the same time I'm trying to find out if there are enough people interested in buying the bare board with documentation directly from me.

All operating software - unless you write your own - should be obtained from the originel software companies mentioned earlier or from their licensed dealers.

Although all values and position numbers are printed on the board, only experienced hobbyists or professionals should tackle the assembly.

Gunter Jansen
Unter Oelbach 1
D-5090 Leverkusen 3
W. Germany
Phone: (0)2171-30676

BIT BUCKET

OS9 SIG ON COMPUSERVE

An OS9 Special Interest Group is available on CompuServe; a nation wide network.

A Special Interest Group (SIG) is a unique utility on CompuServe that enables users with similiar interest to share ideas. There are three (3) major areas to the SIG.

1. Mail or message switching.
2. Conferencing.
3. Data Bases.

Message switching is the exchange of information from one user to all other users. Messages are normally read, written, and stored in serial fashion with the oldest at the "bottom of the stack". Various methods of reading these messages are available. Some of the options are:

1. Read marked messages.
   Messages sent to you, or marked by you.
2. Read selective message by subject.

Other options are available, which the CompuServe publication for Special Interest Groups covers in more detail.

Conferencing:

Conferencing permits members of the SIG to have live interaction conversations wtih one another to discuss various topics.

Data Bases:

Data Bases are divided into sections. This permits easy maintenance and accessibility. To start, there are four data bases. These can be added to or modified as needed. The four data bases are:

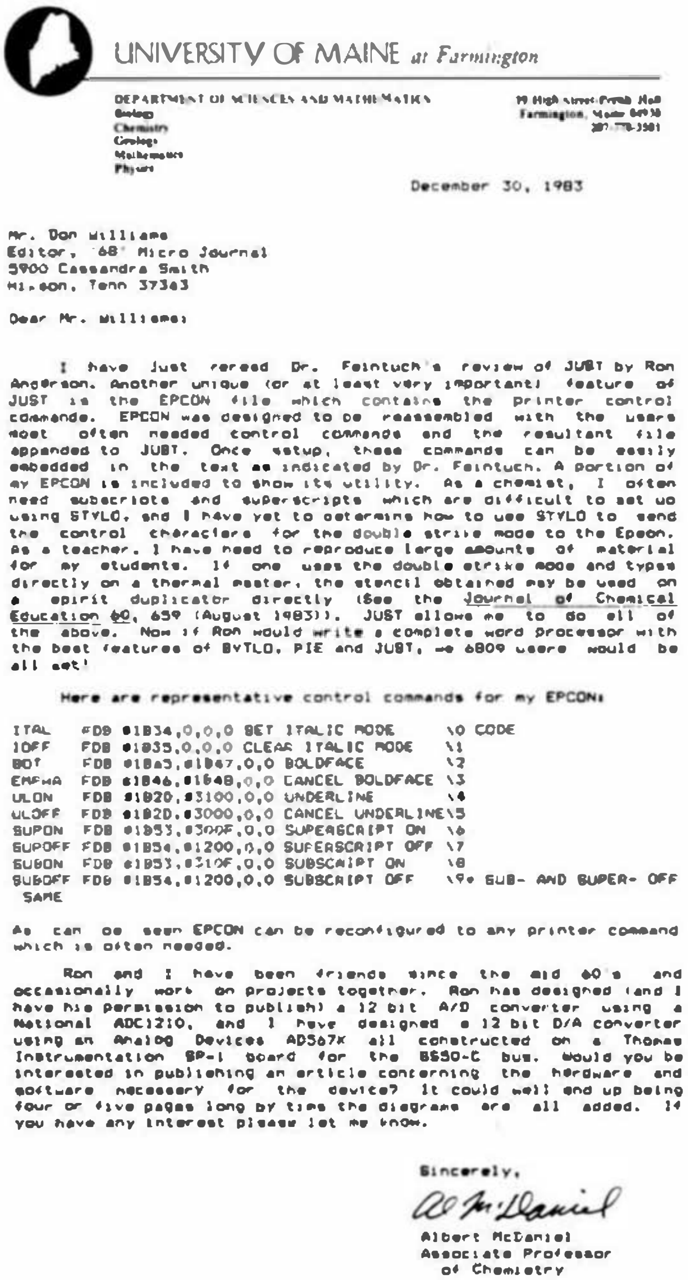
1. Data Base 0: General information.
   (System information, ect.)
2. Data Base 1: Basic09.
3. Data Base 2: "C" section.
4. Data Base 3: Pascal.

As stated before, the operation of the SIG is explained in greater detail in the publication "Special Interest Group Instructions" avaiable form CompuServe. One can join CompuServe through their local computer store or by contacting CompuServe at:

CompuServe
Information Service Division
5000 Arlington Centre Blvd.
Columbus Ohio 43220

Any comments or suggestions will be welcomed.

Jim Bellomo
76703,467
System Operator



UNIVERSITY OF MAINE *at Farmington*

DEPARTMENT OF SCIENCES AND MATHEMATICS
Biology
Chemistry
Geology
Mathematics
Physics

19 High Street-Preble Hall
Farmington, Maine 04938
207-778-3501

December 30, 1983

Mr. Don Williams
Editor, '68' Micro Journal
5900 Cassandra Smith
Hixson, Tenn 37343

Dear Mr. Williams:

I have just reread Dr. Feintuch's review of JUST by Ron Anderson. Another unique (or at least very important) feature of JUST is the EPCON file which contains the printer control commands. EPCON was designed to be reassembled with the users most often needed control commands and the resultant file appended to JUST. Once setup, these commands can be easily embedded in the text as indicated by Dr. Feintuch. A portion of my EPCON is included to show its utility. As a chemist, I often need subscripts and superscripts which are difficult to set up using STYLO, and I have yet to determine how to use STYLO to send the control characters for the double strike mode to the Epson. As a teacher, I have need to reproduce large amounts of material for my students. If one uses the double strike mode and types directly on a thermal master, the stencil obtained may be used on a spirit duplicator directly (See the Journal of Chemical Education 60, 659 (August 1983)). JUST allows me to do all of the above. Now if Ron would write a complete word processor with the best features of STYLO, PIE and JUST, we 6809 users would be all set!

Here are representative control commands for my EPCON:

```
ITAL   FDB $1B34,0,0,0 SET ITALIC MODE     \0 CODE
IOFF   FDB $1B35,0,0,0 CLEAR ITALIC MODE   \1
BOT    FDB $1BA3,$1B47,0,0 BOLDFACE        \2
EMPHA  FDB $1B46,$1B48,0,0 CANCEL BOLDFACE \3
ULON   FDB $1B2D,$3100,0,0 UNDERLINE       \4
ULOFF  FDB $1B2D,$3000,0,0 CANCEL UNDERLINE\5
SUPON  FDB $1B53,$300F,0,0 SUPERSCRIPT ON  \6
SUPOFF FDB $1B54,$1200,0,0 SUPERSCRIPT OFF \7
SUBON  FDB $1B53,$310F,0,0 SUBSCRIPT ON    \8
SUBOFF FDB $1B54,$1200,0,0 SUBSCRIPT OFF   \9; SUB- AND SUPER- OFF
 SAME
```

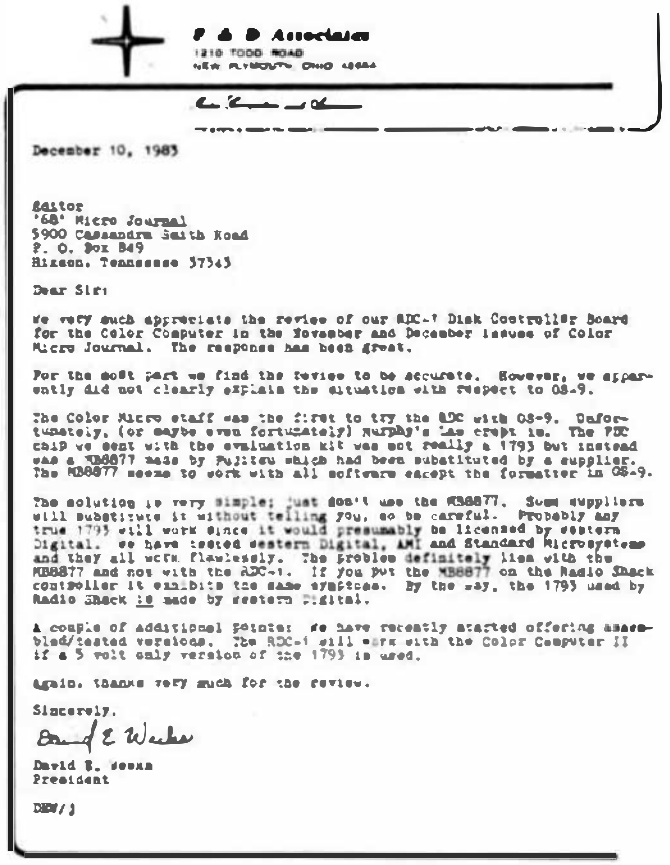
As can be seen EPCON can be reconfigured to any printer command which is often needed.

Ron and I have been friends since the mid 60's and occasionally work on projects together. Ron has designed (and I have his permission to publish) a 12 bit A/D converter using a National ADC1210, and I have designed a 12 bit D/A converter using an Analog Devices AD567K all constructed on a Thomas Instrumentation SP-1 board for the SS50-C bus. Would you be interested in publishing an article concerning the hardware and software necessary for the device? It could well end up being four or five pages long by time the diagrams are all added. If you have any interest please let me know.

Sincerely,

Al McDaniel

Albert McDaniel
Associate Professor
of Chemistry



P & D Associates
1210 TODD ROAD
NEW PLYMOUTH, OHIO 43604

December 10, 1983

Editor
'68' Micro Journal
5900 Cassandra Smith Road
P. O. Box 849
Hixson, Tennessee 37343

Dear Sir:

We very much appreciate the review of our ADC-1 Disk Controller Board for the Color Computer in the November and December issues of Color Micro Journal. The response has been great.

For the most part we find the review to be accurate. However, we apparently did not clearly explain the situation with respect to OS-9.

The Color Micro staff was the first to try the ADC with OS-9. Unfortunately, (or maybe even fortunately) Murphy's law crept in. The FDC chip we sent with the evaluation kit was not really a 1793 but instead was a MB8877 made by Fujitsu which had been substituted by a supplier. The MB8877 seems to work with all software except the formatter in OS-9.

The solution is very simple; just don't use the MB8877. Some suppliers will substitute it without telling you, so be careful. Probably any true 1793 will work since it would presumably be licensed by Western Digital. We have tested Western Digital, AMI and Standard Microsystems and they all work flawlessly. The problem definitely lies with the MB8877 and not with the ADC-1. If you put the MB8877 on the Radio Shack controller it exhibits the same symptoms. By the way, the 1793 used by Radio Shack is made by Western Digital.

A couple of additional points: We have recently started offering assembled/tested versions. The ADC-1 will work with the Color Computer II if a 5 volt only version of the 1793 is used.

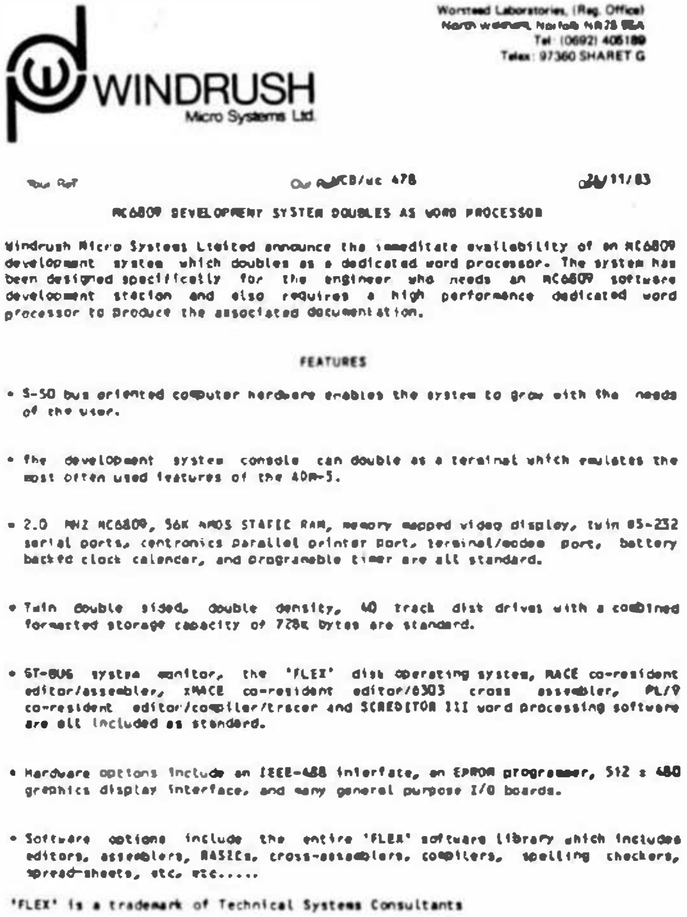
Again, thanks very much for the review.

Sincerely,

David E. Weeks

David E. Weeks
President

DEW/j



WINDRUSH
Micro Systems Ltd

Worstead Laboratories, (Reg. Office)
North Walsham, Norfolk NR28 9SA
Tel: (0692) 405189
Telex: 97360 SHARET G

Your Ref　　Our Ref MCB/uc 478　　Date 24/11/83

MC6809 DEVELOPMENT SYSTEM DOUBLES AS WORD PROCESSOR

Windrush Micro Systems Limited announce the immediate availability of an MC6809 development system which doubles as a dedicated word processor. The system has been designed specifically for the engineer who needs an MC6809 software development station and also requires a high performance dedicated word processor to produce the associated documentation.

FEATURES

- S-50 bus oriented computer hardware enables the system to grow with the needs of the user.
- The development system console can double as a terminal which emulates the most often used features of the ADM-5.
- 2.0 MHZ MC6809, 56K NMOS STATIC RAM, memory mapped video display, twin RS-232 serial ports, centronics parallel printer port, terminal/modem port, battery backed clock calender, and programable timer are all standard.
- Twin double sided, double density, 40 track disk drives with a combined formatted storage capacity of 728K bytes are standard.
- ST-BUG system monitor, the 'FLEX' disk operating system, RACE co-resident editor/assembler, XRACE co-resident editor/6303 cross assembler, PL/9 co-resident editor/compiler/tracer and SCREDITOR III word processing software are all included as standard.
- Hardware options include an IEEE-488 interface, an EPROM programmer, 512 x 480 graphics display interface, and many general purpose I/O boards.
- Software options include the entire 'FLEX' software library which includes editors, assemblers, BASICs, cross-assemblers, compilers, spelling checkers, spread-sheets, etc, etc.....

'FLEX' is a trademark of Technical Systems Consultants

The Best 30 Pin Floppy Controller You Can Buy!

The ELEKTRA Super Floppy Controller

Run:

5″

8″

Single Sided

Double Sided

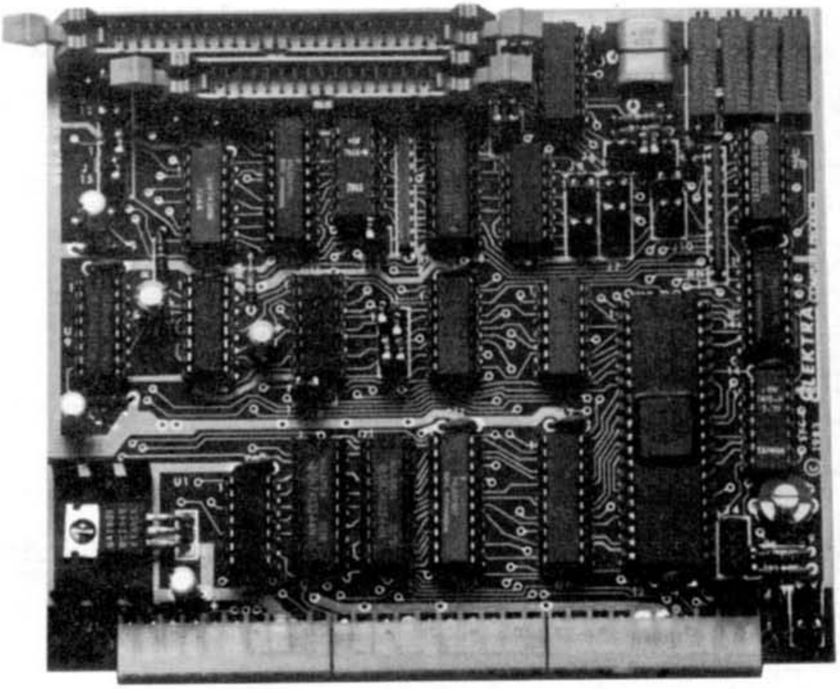
Single Density

Double Density

5″ and 8″
Mixed Systems

1 MHz

2 MHz



(8″ Double Density must run at 2 MHz)

Phone
AAA Chicago Computer Center
Technical consultation available most weekdays from
4 p.m. to 6 p.m. C.S.T.
(312) 459-0450
120 Chestnut Lane, Wheeling, IL 60090

See our catalog and ordering information on the next page to your right.

HELIX

64K 6809 Computer	$2395.00	64K 68008 Computer	$2495.00
256K 6809 Computer	2895.00	256K 68008 Computer	2995.00

Other systems available

20 Megabyte 5" add on Winchester System	$2595.00
64K CMOS Static memory board with battery backup	395.00
DMA 5" and 8" Floppy Controller with built in Winchester controller I/O	695.00
DMA 5" and 8" Floppy Controller $495.00 — 6809 CPU Board	495.00
68008 board for SS-50 595.00 — CP/M-68K	350.00

▶ Need FLEX, UniFLEX, OS-9 Level I, or OS-9 Level II? We have a system for you!

▶ **ELEKTRA COMPUTER CABINET** THE LARGEST SS-50 COMPUTER CABINET AVAILABLE! Made of heavyweight 0.090" thick aluminum. Interior is 18-1/2" wide by 21-7/8" deep by 8-3/4" high. Heavy duty A.C. line cord. A.C. fuse holder. EMI filter. Fan with filter. Back panel has 10 cutouts for 'D' type data connectors. Front panel has key on/off power switch, 2 illuminated push button switches (Reset and NMI/Abort), and two cutouts for 5-1/4" disk drives $250.00

Filler Plate for 5-1/4" drive opening $10.00 — Fan Filter $10.00

POWER SUPPLY Highest quality linear power supply CONSERVATIVELY rated at 15a @ 8v, 3a @ 16v, 3a @ -16v. 3 primary inputs for light, rated, and heavy loading.
220v Version $200.00 — 110v Version $175.00

DISK REGULATOR BOARD WITH CABLES Standard version for 2 floppy drives $50.00
Heavy duty version for 1 Winchester drive and 1 floppy drive $75.00

AUXILIARY POWER SUPPLY to power second Winchester drive $125.00

▶ **ENGINEER'S "FUN BOX" BY ELEKTRA** Computer cabinet with high quality 10 amp power supply, EMI filter, fan. Large enough to hold the standard size SS-50 type motherboard $225.00

ELEKTRA UNIVERSAL SS-50/SS-30C MOTHERBOARD Heavyweight 0.125" thick, 18" long by 9" wide. 11 memory (50 pin) slots 8 I/O (30 pin) slots. Complete address decoding and selection, as well as extended address capability, for I/O slots. Choice of 4, 8, or 16 addresses per I/O slot. 1" spacing between all memory and I/O slots. On board baud rate generator with low and high ranges providing jumper selectable rates of 75 through 38,400 for each of the five baud rate lines. Slow device circuitry permitting 1 MHz 30 pin disk controllers to run with 2MHz 50 pin CPU boards

Mounting hardware	$5.00	Bareboard w/documentation	$80.00
Kit w/gold connectors	$320.00	Assembled w/gold connectors	$380.00
Kit w/tin connectors	$240.00	Asembled w/tin connectors	$300.00

ELEKTRA CHASSIS Includes cabinet, 110v power supply, power supply cables, standard disk regulator board with power cables, motherboard with gold square pin connectors, assembled and tested $850.00

ELEKTRA CPU 8/9 Use either the 6802 or 6808 (to run 6800 software) or 6809. Has provision for up to 3 2716 Eproms, 1K scratchpad, MC6840 triple timer, and an optional baud rate generator providing baud rates from 110 through 38,400 baud in two user selectable ranges. Versions of OS-9 level 1 are available
Bareboard $50.00 — Kit $225.00 — Assembled $275.00
Optional Baud Rate Generator $25.00

ELEKTRA DPS DUAL PORT SERIAL CARD Fits the standard 30 pin SS-50 bus I/O slot. Can be configured for 4 or 16 addresses per port. RTS, CTS, DTR, DCD, IRQ, FIRQ/NMI, and baud rate can be appropriately implemented for each port.
Bareboard $20.00 — Kit $60.00 — Assembled $80.00
Cable with mounting hardware (two needed per board) — Each $25.00
Cable — Each 20.00 — Mounting hardware per cable $5.00

ELEKTRA DPP DUAL PORT PARALLEL CARD Fits the standard 30 pin SS-50 bus I/O slot. Can be configured for 4 or 16 addresses per I/O slot. The direction of the TTL buffers can be controlled by either on board jumper connectors or by a signal from the peripherals. The interrupt request line for each port may be individually jumpered to either the IRQ or FIRQ/NMI bus line.
Bareboard $20.00 — Kit $50.00 — Assembled $80.00
Cable with mounting hardware (two needed per board) — Each $25.00
Cable — Each 20.00 — Mounting hardware per cable $5.00

ELEKTRA 64K STATIC RAM/ROM MEMORY BOARDS with gold connectors (tin available) Assembled and tested. With 56K RAM $269.00 — With 64K RAM $299.00

▶ **ELEKTRA UNIVERSAL SUPER FLOPPY CONTROLLER** THE BEST 30 PIN FLOPPY DISK CONTROLLER THAT YOU CAN BUY! Controls up to four 5-1/4" drives and four 8" drives for a total of eight system drives. (FLEX system limit is four drives.) Single density or double density, 1MHz or 2MHz, 6800 or 6809. (Double density 8" must be at 2MHz; all other combinations of performance are possible.) Analog phase locked loop data separators with separate adjustments for 5" and 8" drives. Analog write precompensation circuit with separate adjustments for 5" and 8" drives. Designed to meet the data hold requirements of Western Digital floppy controller IC. Assembled and tested $275.00
Disk with drivers and formatting utilities (Specify 6800/9, FLEX/OS-9) 30.00

▶ **ELEKTRA WINCHESTER SYSTEMS** THE BEST WINCHESTER SYSTEMS THAT YOU CAN BUY! Has automatic error detection and CORRECTION of up to 11 bit burst errors. SS-50 bus, extended addressing capabilities, DMA, on board sector buffer, drivers included for 6809 FLEX or OS-9. Specify whose version of FLEX that you are using. Drivers for FLEX2 (6800) are available for an additional $100.00. Price includes host interface, controller, drive(s), and cables

12 Megabyte single drive sys	$2295.00	24 Megabyte dual drive sys	$3595.00
19 Megabyte single drive sys	$2995.00	38 Megabyte dual drive sys	$4695.00

(19 Megabyte drives are the largest that can be supported by FLEX)

ELEKTRA HD-5 Cabinet for dual 5 1/4" floppy drives with power supply, line cord, fuse, power switch, and power cables to drives 150.00

▶ **ELEKTRA HD-5W** As above but with EMI filter, fan, and heavy duty power supply. Powers 1 floppy and 1 Winchester 199.00

5' ribbon cable for dual outboard 5-1/4" disk drives	40.00
2' ribbon cable for dual inboard 5-1/4" disk drives	35.00
Custom cables available	Phone

ELEKTRA HD-8 Dual drive cabinet, EMI filter, fan, power supply, and power supply cables for 8" drives 350.00
8' ribbon cable for dual 8" disk drives 45.00

▶ **ELEKTRA 30 PIN PROTOTYPING BOARD**	20.00
▶ **ELEKTRA 50 PIN PROTOTYPING BOARD**	40.00
GOLD 10 PIN CONNECTORS (Specify male with square pins or female)	1.50
TIN 10 PIN CONNECTORS (Specify male with square pins or female)	.50

ELEKTRA is a trademark of AAA Chicago Computer Center
FLEX and **UniFLEX** are trademarks of Technical Systems Consultants, Inc
HELIX is a trademark of Hazelwood Computer Systems
OS-9 and **BASIC09** are trademarks of Microware Systems Corp

Dealer for ELEKTRA, HELIX, SSB, SWTPC, Microware Systems Corp., and Technical Systems Consultants, Inc.

AAA CHICAGO COMPUTER CENTER
120 CHESTNUT LANE • WHEELING, IL 60090 (312) 459-0450
Technical consultation available 4 PM to 6 PM most weekdays. Closed evenings and weekends

TERMS Minimum order $20.00. Shipping and handling estimates within the Continental U.S., add 3% (MINIMUM $2.50). Illinois residents add 6% sales tax. We will refund your overestimated shipping and handling charges. Foreign shipping and handling, add 10% (MINIMUM $10.00). Foreign orders must be prepaid in U.S. dollars. Checks must be drawn on a U.S. bank. Heavy foreign items will be shipped air freight collect. Please Phone between 4 PM and 6 PM weekdays if questions arise regarding shipping fees. Master Charge, Visa, and American Express honored.

Our apology. We are not staffed to answer technical inquiries through the mail. Please phone for technical help during the hours indicated above. The too frequent changing of our inventory and prices makes it uneconomical to publish a catalog. Our ads are intended to serve that purpose. Prices and inventory are subject to change without advance notice.

ELEKTRA™ SOFTWARE (All of our software is copyrighted and all rights are reserved. Source is either supplied or optionally available at extra cost so that the purchaser can modify our programs for his own use. Licensing, however, is required for commercial resale.)

SUPER MODEM PROGRAM Single character commands. No interrupts required. Transmit manually or transmit disk files (text) of any length to distant computer. Receive and save disk files (text) on local disk system. X-on/X-off supported. Tested for full duplex at speeds up to 9600 baud. Half duplex option. Echo option. Replaces CR with CR/LF (user option). Slow disk file transmit option.
Please specify 6800 or 6809, SSB or FLEX™, 5" or 8"
Instruction Manual and disk with both source and object code $75.00

STANDARD MODEM PROGRAM Same as Super Modem Program above but without ECHO option, CR/LF for CR option, slow disk file transmit option, nor X-on/X-off option. Specify 6800 or 6809
Manual with instructions, source listing, and flow chart 10.00

▶ **OS-9 Configurable Modem Program** (Sorry, source is not available) 100.00

ORDER — WRITE UP COMPUTER PROGRAM Screen oriented write up form with cursor editing, disk save and load, printer command using easily available universal print-out forms. Phone for more details. Available for 6809 FLEX $100.00

ALL IN ONE
Editor — Text Processor — Mailing Labels — Mailing Lists — Multiple Form Letters
Use any CRT terminal and printer — Best Package For The Money Anywhere!

Specify 6800 or 6809, SSB or FLEX™, 5" or 8"	75.00
Printed source listing is available for an additional	35.00
All-In-One, Write'n spell, and Spell'n Fix package	250.00

Software by Technical Systems Consultants, Inc.	FLEX(TM) Adapt. Guide	FLEX(TM) Source (List)	FLEX(TM) Source (Disk)	FLEX(TM) Man. Only	FLEX(TM) Object w/Man.	UniFLEX(TM) Add. Man.	UniFLEX(TM) Man. Only	UniFLEX(TM) Object w/Man.
Gen DOS /Edit & ASMB	25	—	—	50	250	—	—	—
SWTPC DOS w/Edit & ASMB		—	—	25	150	40	100	550
Advanced Programmers Guide		—	—	25	—	—	—	—
Editor		100	250	25	50	—	—	—
Assembler		150	250	25	50	—	—	—
Debug		175	250	25	75	—	—	—
Extended Basic		—	—	25	100	20	50	200
Basic Precompiler		—	—	25	50	10	25	150
Sort/Merge		—	—	25	75	20	35	150
Utilities		—	inc.	25	75	10	25	150
Diagnostics		—	—	25	75	—	—	—
Text Processor		150	250	25	75	20	35	150
68000 X-ASMB on 6809		—	—	25	250	20	35	300
Pascal		—	—	50	200	25	50	300
Rel ASMB/Linking Loader		—	—	25	150	20	35	175
6800 X-ASMB on 6809		—	—	—	100	—	—	—
Cobol		—	—	—	—	30	75	750
Fortran 77		—	—	—	—	5	65	450

Software by Microware Systems Corp.	Run-Time Package	Update	Source	Manual Only	Object w/Man.
OS-9™ Level One Operating System		75.00	400.00	40.00	200.00
OS-9™ Level Two Operating System		75.00	N/A	40.00	500.00
BASIC09™	100.00	75.00	N/A	25.00	200.00
OS-9™ Macro Text Editor			300.00	15.00	125.00
OS-9™ Interactive Assembler			300.00	10.00	125.00
OS-9™ Interactive Debugger (Disk version)			100.00	10.00	50.00
CIS Cobol Compiler	400.00	50.00	N/A	80.00	900.00
Pascal Compiler	100.00	100.00	N/A	40.00	400.00
"C" Compiler		100.00	N/A	40.00	400.00

Microware yearly support service ($200.00 for OS-9 Level 2) 75.00

Special Software

2K 6809 MICROBUG	30.00	4K 6809 HUMBUG	75.00
2K 6800 HUMBUG	40.00	4K 6800 HUMBUG	65.00

Other HUMBUG versions including video versions are available

Spell'n Fix by Peter Stark 178.58 — Write'n Spell by Peter Stark	75.11
All-In-One, Spell'n Fix, and Write'n Spell package	250.00
Dynamite Disassembler	60.00
SUPER SLEUTH Disassembler System ($101.00 for OS-9 version)	99.00

SD/DD DISK DRIVES 30 day guarantee	1 head Tandon	2 heads Tandon	2 heads CDC	1 head MPI	2 heads MPI
5-1/4", 40 tracks	225.00	300.00	300.00	250.00	325.00
5-1/4", 80 tracks	300.00	375.00	375.00	325.00	400.00

Mpi or CDC Service Manual (Specify 40 or 80 track) 25.00
8", 77 tracks DS/DD — Qume DT-8 $650.00 — Remex (Special) 350.00

▶ **SPECIALS**

• U.S. Robotics 300/1200 baud auto dial/auto answer modem	99.00
• Same as above but without self test and diagnostics	389.00
• U.S. Robotics 1200 baud direct connect auto answer modem	349.00
• Hazeltine 1420 CRT terminal (new)	375.00
• SSB BFD Floppy Disk Controller (Version 3) Run FLEX or SSB DOS	100.00
• SWTPC 4K Memory $15.00 — MP-B $40.00 — MP-A with MP-C	35.00
• High speed tape reader 50.00 — AC-30 $40.00 — MP-S	60.00
• 300 Baud acoustic modem 129.00	
• 10 ELEKTRA DD 8" Disks 35.00 — 10 5" DD Disks in hard box	5.00
• TI 810 printer w/lower case and full vertical forms control	1200.00
• Microtime II Calendar and Clock Board (Assembled)	60.00

GIMIX CLEARANCE SALE	LIST	OUR PRICE		LIST	OUR PRICE
#5 6809 Plus CPU Bd	578.05	475.00	6800 CPU board	2 4.03	100.00
Cable (Ser or Par I/O)	24.95	20.00	Motherboard		200.00
Double disk reg. card	68.22	50.00	8 Port Serial I/O Bd.	318.46	250.00
32K memory board		175.00	#28 control w/GMX Flex	328.28	270.00
64K memory board	478.67	450.00	56K memory board		425.00
80 X 24 Video Boards	398.76	250.00	Single prt ser, 1 cable	113.36	90.00
64 X 16 Video Boards	199.71	100.00	Dual prt par, 2 cables	138.32	110.00
16K Mem.Bds w/CMdtreg		145.00	4K PPD PROM Bd and burner		100.00
93L42 RAM chips (2 needed for GIMIX DAT)				each	20.00

SWTPC

DC-4 5-1/4" Disk Controller	230.00	SWTPC 6809 FLEX™ Disk & Man	35.00
MP-S2 Dual Port Serial	120.00	MP-L2 Dual Port Parallel	120.00
MP-N Calculator Board (kit)	54.95	MP-N (assembled)	92.00
MP- 2716 Eprom Programmer	114.50	MP-09 2MHz 6809 CPU Board	295.00

Smoke Signal Broadcasting

DCB-4A Double Density Controller Board for 5" and 8" with DOS	549.00
SSB DOS (Specify 6800 or 6809, BFD or DCB-4A, 5" or 8")	75.00
SE92/SA92-5 (6809 Edit/Assm for DOS)	69.96
SSB Monitor (Specify 6800/6809, $8008/$E008/$F7E8)	75.00
SSB version of FLEX™ (without Editor and Assembler)	150.00
LMB-1A Motherboard	399.00
SCB-69 6809 CPU Board	399.00
PAR-1 Dual Port Parallel Board	89.00
SER-2 Dual Port Serial Board with 2 Cables	129.00
Chief 90 64K Computer System	2195.00

WARNING AAA Chicago Computer Center does not provide repair or diagnostic service for customer assembled kits. AAA Chicago Computer Center does warranty and maintain service for our assembled boards. The customer should carefully take into consideration the small differential separating our kit and assembled prices when making his choice of purchase.

We have introduced our line of computer equipment with the purpose of offering the highest quality of components possible at affordable prices. These products are intended for OEM applications where it is the responsibility of the purchaser to integrate these components with suitable memory, disk controllers, drives, and software along with I/O terminals to form working computer systems.

MICROWARE. PRESS RELEASE

September 15, 1983
For Immediate Release
Contact: Jody Hall, 515-279-8844

ENTERTAINMENT PACK I NOW AVAILABLE FOR OS-9

Microware has release "Entertainment Pack I, a collection of programs written in Basic09 for the OS-9 Operating System. The package consists of games and other interesting programs that are not only entertaining but serve as excellent instructional examples of Basic09 programming techniques. All programs include complete Basic09 source files and can be easily edited to run on standard alphanumeric or graphics terminals.

The programs included in the package are:

Blkjck - A Vegas-rules blackjack game.
Clk - graphical display of a wall clock on your terminal.
Dogs - Greyhound racing with simulated graphics on standard terminals.
Eliza - Basic09 version of the famous artificial intelligence simulation of natural language dialogue with a psychiatrist.
Haiku - Program that creates original "haiku", flowing Japanese prose.
Quest - a mini-"Adventure" game
Rats - find you way out of a computer-generated maze - from a rat's point of view.
Towers - a graphical display of the solution to the "Tower of Hanoi" puzzle.

Although all other issues including November's have arrived OK, I guess the October '83 issue of 68 Micro Journal got lost in the post. I can't do without my monthly ration of helpful hints so please send another copy.

I enclose details of a method of reading 35/40 track disks on a system fitted only with 80 track drives. There has been talk in the journal that this might be possible but up to now you haven't published details of how to do it.

Now anyone could have a system with only 80 track drives. It hasn't been practical before because most software is supplied on 'Standard' 35 track disks.

MODIFIED GENERAL FLEX DRIVERS
To read/write 40 track disks on 80 track drives

I know it shouldn't work in theory - See Ron Andersons comments in the MAR 83 issue - but it does for me! just in case,once a disk is formatted to 80 tracks I keep it to that format.

With the two new routines detailed below incorporated in the General FLEX disk drivers in place of the existing SEEK and DRV routines the drives will behave quite normally. When you wish to read or write to a 40 track disk simply put the disk in drive 1 but tell FLEX its in drive 2. For example, entering LIST 2.TEST would list the file TEST.TXT from a 40 track disk in drive 1.

I don't think 68 MICRO would publish the FLEX drivers in full but luckily we only need to change two routines as follows:

```
* NEW SEEK ROUTINE

SEEK  STB SECREG   Set sector
      LBSR DEL28   let 1771 digest this
      TST FLG40
      BEQ SEEK1    skip if flag not set
      ASL TRKREG   double the No. in track reg
      LBSR DEL28
      ASLA         Double regd track count in A
SEEK1 STA DATREG   Go find a track
      LBSR DEL28
      LDA £01B     Set up Seek Cmd
      STA COMREG   tell 1771
      LBSR DEL28
      BSR WAIT
      TST FLG40
      BEQ SEEK2    skip if flag not set
      LSR TRKREG   else halve the No. in track reg
      LBSR DEL28
SEEK2 BITB £010
      RTS
FLG40 FCB 0
*  NEW DRV ROUTINE

DRV   CLR FLG40    preset flag each call
      LDA 3,X      get drive No.
      CMPA £3
      BLS DRV2     OK if 3 or less
      CLRA         else set drive 0
DRV2  BSR FNDTRK   point to old drive track store
      LDB TRKREG   get current track
      STB 0,X      save it
      STA CURDRV   set new drive
      BSR FNDTRK   point to new drive track store
      LDB 0,X      get track in B
      STB TRKREG   tell track reg
      BSR DEL28
      CMPA £2      is new drive No.2
      BNE ASIS     if not leave as it is
      LSRA         Change 2 into 1
       TA FLG40    set the flag
ASIS   TA DRVREG   tell drive reg
      BSR DEL28
      BRA OK
```

Labels of external routines are as in the General FLEX documentation. All the other driver routines are unaltered.

Barrie R Smith
1 Belle Vue Rd
Herne Bay,Kent
England.

```
*  CLOCK (RENAMED "CLOCK" FROM "TIME" WHEN MODIFIED)
*
*  This program prints the day, date, and time on the
*  console.  It gets the day, date, month, and time
*  from the 58167 real time clock.  The year comes
*  from a 5-byte area in the CPU's CMOS RAM scratchpad
*  where it is stored.  This program uses external
*  routines only from GMXBUG-09 and is therefore
*  independent of any operating system.
*
*  Being independent of any operating system is its
*  great benefit.  It can even be accessed by
*  interrupt service routines and software which
*  modifies your operating system (and some do!).
*  Being memory resident, it performs quickly enough
*  to be used as a time stamp generator for other
*  programs.  In this application, it would have to
*  have its output redirected.  I use it to generate
*  a time line on the 25th line of my Heath H19
*  terminal at either one second or one minute
*  intervals.
*
*  This program is accessed with a JSR to its base.
*  address.  This version is installed at $E440 in a
*  reconfigured Gimix 6809+.  It can be relocated
*  elsewhere by changing only the ORG address and
*  reassembling it.
*
*  CLOCK is a modification of the TIME program.  The
*  TIME program was written by:
*
*               GIMIX, INC.
*               1337 WEST 37TH PLACE
*               CHICAGO, ILLINOIS  60609
*               (312) 927-5510
*  Gimix has generously given permission for this version
*  to be released into the public domain.
*
*  Modification Record:
*
*  MODIFIED FOR RELOCATION AT ORG $F000 AND AS A
*  SUB-PROGRAM MODULE:                         21-aug-83
*               Philip C. Munn
*               201 Netherfield
*               Comstock Park, MI.  49321
*
*  MODIFIED TO BE INDEPENDENT OF FLEX, USE GMXBUG-09
*  ROUTINES, AND RESIDE IN CMOS MEMORY:
*              Philip C. Munn                  23-OCT-83
*
*  Modified for installation in unused CMOS scratchpad
*  at $E440 - $E842:
*              Philip C. Munn                  24-Oct-83
* ..................................................
*
*  EQUATES
*
OUTCHR   EQU      $F80A               SINGLE CHAR OUT
PSTRNG   EQU      $F810               OUTPUT CR/LF & STRING
PCRLF    EQU      $F80E               OUTPUT CR/LF
PSPACE   EQU      $F816               PRINT A SPACE
SYSCLK   EQU      $E220               CLOCK BASE ADDRESS
YEAR     EQU      $E43B               YEAR BUFFER
* ..................................................
*
*  START OF PROGRAM
*
         ORG      $E440

CLOCK    CLR      FLAG                CLEAR AM/PM FLAG
         LDX      #TIMLOC             POINT TO TIME AREA IN RAM AT $F1F0
         JSR      CLOKRD              READ CLOCK
         LDY      #TIMLOC             POINT TO START OF STRING AT $F1F0

         LDB      0,Y+                GET DAY OF WEEK FROM $F1F0
         DECB                         ADJUST FOR LOOKUP
         CMPB     #6                  VALID DAY?
         LBHI     ERROR               NO:  ERROR
         ASLB                         SHIFT FOR 2 BYTE TABLE
         LDX      #DYLKUP             POINT TO DAY LOOK UP TABLE
         LDX      0,X                 POINT TO STRING
         LBSR     PDATA               PRINT DAY OF WEEK

         LDB      0,Y+                GET MONTH FROM $F1F1
         DECB                         ADJUST FOR LOOKUP
         CMPB     #10                 IS IT 2 DIGITS?
         BLO      TIME1               NO:  DON'T ADJUST
         CMPB     #$F                 VALID TIME?
         LBLO     ERROR               NO:  ERROR CLOCK NOT SET
         SUBB     #$6                 ADJUST FOR HEX OFFSET
         CMPB     #11                 VALID MONTH?
         LBHI     ERROR               NO:  ERROR
```

```
TIME1   ASLB                    SHIFT FOR 2 BYTE TABLE
        LDX     #MNLKUP         POINT TO MONTH LOOK UP TABLE
        LDX     B,X             POINT TO STRING
        LBSR    PDATA           PRINT MONTH

        LDA     0,Y+            GET DATE FROM $F1F2
        BSR     OUTBYT          OUTPUT DATE

        LDA     #',             GET COMMA
        JSR     [OUTCHR]        PRINT COMMA
        JSR     [PSPACE]        PRINT A SPACE
        LDX     #YEAR           POINT TO YEAR BUFFER
        LBSR    PDATA           PRINT YEAR
        JSR     [PSPACE]        SKIP A SPACE
        JSR     [PSPACE]        SKIP A SPACE
        JSR     [PSPACE]        SKIP A SPACE
        JSR     [PSPACE]        SKIP A SPACE

        LDA     0,Y+            GET HOURS FROM $F1F3
        CMPA    #$23            ILLEGAL TIME?
        LBHI    ERROR           YES:  PRINT ERROR
        PSHS    A               PRESERVE A
        ANDA    #$F             MASK OUT HIGH ORDER NYBBLE
        EORA    #$C             ILLEGAL TIME?
        LBEQ    ERROR           YES:  PRINT ERROR
        PULS    A               RESTORE A
        TSTA                    MIDNIGHT?
        BNE     NOTMID          NO:  THEN DON'T ADJUST
        LDA     #$12            MIDNIGHT
        BRA     TIME6           AND PRINT HOURS

NOTMID  CMPA    #$12            AM OR PM?
        BLO     TIME6           AM:  FLAG ALREADY CLEARED
        BEQ     TIME5           IF NOON DON'T ADJUST
        CMPA    #$21            22 OR 23 HUNDRED HOURS?
        BHI     ADJUST          YES:  DON'T DO OFFSET
        CMPA    #$20            20:00 HOURS?
        BLO     ADJUST          YES:  DON'T DO OFFSET
        SUBA    #6              SUBTRACT OFFSET FOR NON-HEX

ADJUST  SUBA    #$10            ADJUST HIGH ORDER NYBBLE
        SUBA    #2              ADJUST LOW ORDER NYBBLE

TIME5   INC     FLAG            SET TO PM

TIME6   BSR     OUTBYT          OUTPUT HOURS
        LDA     #':             GET COLON
        JSR     [OUTCHR]        PRINT COLON
        LDA     0,Y+            GET MINUTES
        BSR     OUTBYT          OUTPUT MINUTES
        LDA     #':             GET COLON
        JSR     [OUTCHR]        PRINT COLON
        LDA     0,Y+            GET SECONDS
        BSR     OUTBYT          PRINT SECONDS
        TST     FLAG            AM OR PM?
        BEQ     AM              IF AM POINT TO AM MESSAGE
        LDX     #PMMSG          POINT TO PM MESSAGE
        BRA     MSGOUT          BRANCH AROUND PM SET

AM      LDX     #AMMSG          POINT TO AM MESSAGE

MSGOUT  JSR     PDATA           PRINT PM OR AM

FINISH  RTS                     RETURN TO CALLING ROUTINE
* ..................................................
*
*  OUTPUT BYTE
*
OUTBYT  PSHS    A               SAVE 'A' FOR LATER
        RPT 4
        LSRA                    GET INTO LS NYBBLE
        BSR     OUTNYB          OUTPUT NYBBLE
        PULS    A               RETRIEVE 'A'
        ANDA    #$0F            MASK OUT MS NYBBLE

OUTNYB  ADDA    #$30            ADD FOR DECIMAL
        JSR     [OUTCHR]        OUTPUT IT
        RTS                     RETURN TO MAINLINE
* ..................................................

WEKTAB  EQU     *
MON     FCC     /MONDAY   /
        FCB     $04
TUES    FCC     /TUESDAY   /
        FCB     $04
WED     FCC     /WEDNESDAY   /
        FCB     $04
THURS   FCC     /THURSDAY   /
        FCB     $04
FRI     FCC     /FRIDAY   /
        FCB     $04
SAT     FCC     /SATURDAY   /
        FCB     $04
SUN     FCC     /SUNDAY   /
        FCB     $04

MONTAB  EQU     *
JAN     FCC     /JANUARY /
        FCB     $4
FEB     FCC     /FEBRUARY /
        FCB     $4
MARCH   FCC     /MARCH /
        FCB     $4
APRIL   FCC     /APRIL /
        FCB     $4
MAY     FCC     /MAY /
        FCB     $4
JUNE    FCC     /JUNE /
        FCB     $4
JULY    FCC     /JULY /
        FCB     $4
AUGUST  FCC     /AUGUST /
        FCB     $4
SEPT    FCC     /SEPTEMBER /
        FCB     $4
OCT     FCC     /OCTOBER /
        FCB     $4
NOV     FCC     /NOVEMBER /
        FCB     $4
DEC     FCC     /DECEMBER /
        FCB     $4

DYLKUP  FDB     MON,TUES,WED,THURS,FRI,SAT,SUN

MNLKUP  FDB     JAN,FEB,MARCH,APRIL,MAY,JUNE
        FDB     JULY,AUGUST,SEPT,OCT,NOV,DEC
* ..................................................

*  START OF 'CLOKRD'
*
CLOKRD  LDY     #SYSCLK+5       POINT TO CLOCK ($E220+5=$E225)
        LDA     0,Y             GET DAY OF THE WEEK FROM $E225
        STA     0,X+            STORE AT $F1F0 AND BUMP POINTER
        LDA     2,Y             GET MONTH FROM $E227
        STA     0,X+            STORE AT $F1F1 AND BUMP POINTER
        LDA     1,Y             GET DAY OF THE MONTH FROM $E226
        STA     0,X+            STORE AT $F1F2 AND BUMP POINTER
        LDB     #$05            NUMBER OF BYTES TO TRANSFER

CLOKR1  LDA     0,-Y            GET VALUE FROM $E224,3,2,1,0 AND
*                                 DECRIMENT POINTER
        STA     0,X+            STORE AT $F1F4,5,6,7,8 AND BUMP
*                                 POINTER
        DECB                    DECRIMENT COUNTER
        BNE     CLOKR1          LOOP TILL DONE WITH 5 BYTES
        LEAX    -8,X            SET X TO POINT TO $F1F0
        TST     SYSCLK+$14      DID IT CHANGE WHILE READING
*                                 SYSTEM CLOCK ($E234)
        BNE     CLOKRD          IF CHANGED WHILE READING, RE-READ
        RTS                     RETURN WHEN DONE
* ..................................................

*  PRINT STRING W/O CR & LF
*
PDATA   LDA     0,X+            GET CHARACTER
        CMPA    #4              END OF STRING?
        BEQ     OUT             YES:  EXIT
        JSR     [OUTCHR]        PRINT IT
        BRA     PDATA           LOOP TILL DONE

OUT     RTS                     EXIT
* ..................................................

ERROR   LDX     #ERRMSG         POINT TO ERROR MESSAGE
        JSR     [PSTRNG]        PRINT IT
        LBRA    FINISH          EXIT

ERRMSG  FCC     /ERROR IN READING TIME,  CLOCK NOT SET/
        FCB     4

AMMSG   FCC     / AM/
        FCB     4

PMMSG   FCC     / PM/
        FCB     4

FLAG    FCB     0
* ..................................................

TIMLOC  EQU     *
        END     CLOCK
```

Ronald W. Anderson
3540 Sturbridge Ct.
Ann Arbor, MI 48105

Nov. 13, 1983

'68' Micro Journal
P.O. Box 849
Hixson, TN 37343

Attention: Don Williams Sr.

Dear Don,

I've found that JUST has a problem running in a CoCo. It is a simple matter of having put the stack in the wrong place. I had located it near the end of memory, and now find that the CoCo uses that area for video drivers. I'd appreciate it if you would send the following patch to the owners of JUST. It can be fixed by changing one byte of the code as follows:

1. Using the GET command of FLEX, load JUST.CMD

   ```
   +++GET JUST.CMD.0
   ```

2. Using the monitor, change the byte at address $0002 from $80 to $3F. (See FLEX manual. Procedure depends on which version of FLEX you have).

3. Using the SAVE utility, save the modified JUST.

   ```
   +++SAVE JUSTNEW.CMD.0,0,124F,0
   ```

4. Now try JUSTNEW on a text file. If you have done everything

correctly it will work properly. After trying it, delete the old JUST.CMD, and rename JUSTNEW.CMD to JUST.CMD.

Incidentally, the patch does not make JUST incompatible with SS-50 bus systems in any way, and JUST will be supplied in this form from now on. The change has moved the "system stack" from $BDFF to $3FFF. JUST might possibly expand in the future to take advantage of a larger special function table for the printer, but $3FFF is undoubtedly far enough above program end at $124F that there will never be a need to move it again. I am truly sorry if anyone has been caused any inconvenience by this problem.

Yours truly,
Ron
Ron Anderson

PRESS INFORMATION

EDITORIAL CONTACTS:

MOSTEK CORPORATION:	TOM NUNAN (214) 466-6315
MOTOROLA INC.:	LOTHAR STERN (602) 994-6316
SIGNETICS CORP.:	MURRAY SHOHAT (408) 746-2178

EXPANDED ARCHITECTURE, NEW CHIP SUPPORT BROADEN VMEbus CAPABILITIES

Phoenix, Arizona, November 3, 1983— Projecting a market size of $750M for VMEbus compatible products by 1988, in a total microcomputer board market of $2.2B, executives of Mostek, Motorola and Signetics/Philips have announced technical agreement on additional key elements of the total VMEsystem Architecture. The newly-defined supporting bus structures include a high-speed memory expansion bus, VMXbus, and a self-arbitrating high-speed serial bus, VMSbus, that may be optionally employed in systems based on the existing VMEbus standard.

These additional buses join the basic VMEbus interconnect that was originally introduced in October 1981, and has since been adopted by over 60 vendors worldwide. As a further measure of broad industry acceptance, the VMEbus is presently being formally standardized by both the IEEE (as P1014) and the IEC (International Electrotechnical Committee, based in Geneva).

The VMEbus and its extension bus structures form the basis of a total VMEsystem Architecture which simplifies integration of complete systems from high-performance 8-, 16-, and 32-bit board level system components. The existing VMEbus interconnect standard provides the basic data transfer bus between major system components, while the VMXbus facilitates expansion of local processor memory with very high performance characteristics. For rapid communication of brief messages between system modules, the VMSbus serial bus uses only two conductors (clock and data) to provide an efficient "party line" between system components.

Mostek, Motorola, and Signetics/Philips have agreed on detailed technical specifications for the new VMXbus and VMSbus, for which documentation has been developed and made available. All three companies have agreed to produce a further high-level specification for the total VMEsystem Architecture, promoting compatibility between VMEsystem products from many vendors. In addition, the companies have indicated that they will be announcing independently-developed products incorporating VMEsystem Architecture during 1984.

Following the custom established with the VMEbus, the companies are placing the new bus specifications and supporting documents into the public domain, with no copyright or licensing requirements. It is believed that this policy will promote widespread availability of VMEsystem-compatible products, as end-users and OEMs increasingly adopt VMEsystem Architecture for their high-performance 16- and 32-bit system products.

Mostek and Signetics/Philips both produce the 16/32-bit M68000 MPU originally developed by Motorola. While not limited in application to 68000-based systems, the VMEsystem Architecture has been designed to provide a set of system-level features desirable for high-performance microcomputer systems where the M68000 MPU family will be used most frequently. Since its introduction, VMEbus has been widely applied in industrial process control, image processing, engineering work-stations, and digital network communications.

Additional bus-support IC designs in the M68000 family have also been announced to facilitate development of VMEsystem-compatible boards. These new ICs include the 68172 VMEbus Controller (E-BUSCON), the 68173 VMSbus Controller (S-BUSCON), and the 68174 Bus Arbiter. Of these, Signetics will develop the 68172 and 68173 designs, while Motorola will develop the 68174 part.

The companies project that samples of these new bus support chips will be available in the second half of 1984. Continuing technical discussions among the three companies are expected to result in definition of additional bus support chips to complement the overall VMEsystem Architecture.

Mike Johnshoy
707 Continental Circle, 1213
Mt View, CA 94040
(415) 967-2048

DISK AND RAM DISK TESTING

With the prices of memory chips falling, and companies like Digital Research Computers offering low cost boards, it is becoming common for 68xx users to have more than 64K of memory. Several schemes are in use to allow this additional memory to be used as a disk drive under FLEX, providing a great speed advantage over the standard 5 1/4 inch floppy. While first getting my own RAM disk into operation, I discovered the discount RAM chips I was using were not all good. The problem then was to find bad chips on four 64K RAM boards. The only means at my disposal was to run one board in my system at a time and run the standard memory diagnostics - SUMTEST, MEMCON, and ROBIT. While these tests did the job, I cringed over the number of times I plugged those boards into and out of my system in the course of debugging.

My solution to this problem is the following utility, DISKTEST.CMD. This utility tests a RAMDISK or a real disk - the program can't tell the differnce since standard FLEX routines are used for reading and writing. The disk to be tested must be formatted first, since this utility reads the System Information Record on track 0, sector 3 to determine the number of tracks and sectors to be tested. Every track and sector on the disk will be tested, and the disk will require reformatting before it can be used again. Bad tracks and sectors are reported and testing continues until the entire disk is tested.

An interesting feature of this utility is the way it is called. If your memory for utility command formats is good, you can pass the parameter required (the disk drive number to be tested) on the command line. If you forget the format required by the command, the utility will prompt you for the information required. It adds a small amount of code, but when your collection of utilities starts to get large it's nice to be able to operate them all without constantly rummaging through documentation.

This utility will not find every problem in memory or on a disk. It does however run very quickly and provides a 'warm feeling' that those cheap chips (or disks) are still hanging in there. I would be interested in any improvements readers may make in improving the ability of this utility to find memory problems. I hope this utility provides a useful start.

```
                    **************************************************
                    *                                                *
                    *  DISK TEST UTILITY                             *
                    *                                                *
                    *  THIS UTILITY TESTS A DISK BY WRITING AN       *
                    *  ALTERNATING PATTERN OF 1's AND 0's TO EACH    *
                    *  SECTOR.  THE DISK TO BE TESTED MUST BE        *
                    *  FORMATTED IN THE FLEX FORMAT BEFORE TESTING.  *
                    *  TESTING DESTROYS ALL DATA ON THE DISK, AND    *
                    *  THE DISK WILL HAVE TO BE REFORMATTED BEFORE   *
                    *  USE.                                          *
                    *                                                *
                    *  SAMPLE USAGE:                                 *
                    *     DISKTEST,2           (Will test Drive 2)   *
                    *                                                *
                    *  WRITTEN BY: MIKE JOHNSHOY                     *
                    *              707 CONTINENTAL CIRCLE, #1213     *
                    *              MT VIEW, CA  94040                *
                    *                                                *
                    *  LAST MODIFIED: 3 DEC 83                       *
                    *                                                *
                    **************************************************

0000 02             PASSES  FCB    2         NUMBER OF TEST PASSES TO BE MADE

                    * FLEX EQUATES

             CC03   TTYEOL  EQU    $CC03
             CD27   NXTCH   EQU    $CD27
             CD3C   OUTHEX  EQU    $CD3C
             CD1E   PSTRNG  EQU    $CD1E
             CD15   GETCHR  EQU    $CD15
             CD24   PCRLF   EQU    $CD24

             DE00   READ    EQU    $DE00     READ A SINGLE SECTOR
             DE03   WRITE   EQU    $DE03     WRITE A SINGLE SECTOR
             DE0C   DRVSEL  EQU    $DE0C     SELECT THE SPECIFIED DRIVE

             CD03   WARMS   EQU    $CD03

C100                        ORG    $C100     FLEX UTILITY SPACE

C100 20   01                BRA    START
C102 01             VN      FCB    1         VERSION NUMBER

                    * CHECK COMMAND LINE FOR DRIVE NUMBER

C103 BD   CD27      START   JSR    NXTCH     GET DRIVE NUMBER FROM CMND LINE
C106 34   02                PSHA             SAVE DRIVE NUMBER
C108 BD   CD27              JSR    NXTCH     INCREMENT LINE SUFFER POINTER
C10B 35   02                PULA             RECOVER DRIVE NUMBER
C10D 81   CC03              CMPA   TTYEOL    WAS IT MISSING ON CMND LINE?
C110 1027 00CE              LBEQ   MENU
C114 81   0D                CMPA   #$0D      WAS IT MISSING ON CMND LINE?
C116 1027 00C8              LBEQ   MENU
C11A 80   30                SUBA   #$30      CONVERT ASCII TO BINARY

                    * SELECT DRIVE NUMBER

C11C B7   C262      SELECT  STAA   DRIVE     SAVE DRIVE NUMBER TO BE TESTED
C11F 8E   C262              LDX    #DRIVE    GET READY FOR CALL TO SELECT
C122 30   1D                LEAX   -3,X
C124 BD   DE0C              JSR    DRVSEL    USE FLEX TO SELECT DRIVE
C127 1025 00C9              LBCS   ERROR

                    * READ FORMATTED DISK TO DETERMINE SIZE

C12B 86   00                LDA    #$00      POINT AT TRACK 0
C12D C6   03                LDB    #$03      POINT AT SECTOR 3
C12F 8E   C368              LDX    #BUFF2    POINT AT BUFFER 2
C132 BD   DE00              JSR    READ      READ THE SYSTEM INFO RECORD
C135 8E   C368              LDX    #BUFF2
C138 A6   88 26             LDA    38,X      GET TRKMAX
C13B B7   C264              STA    TRKMAX
C13E A6   88 20             LDA    32,X      GET SECMAX
C141 B7   C265              STA    SECMAX
C144 A6   88 27             LDA    39,X      GET NUMBER OF SECTORS PER TRACK
C147 B7   C263              STA    SECTRK

                    * INITIALIZE OTHER VARIABLES

C14A 86   55                LDA    #%01010101 INITALIZE TEST PATTERN
C14C B7   C261              STA    TSTPAT
C14F 7F   C260              CLR    PASCNT    CLEAR PASS COUNTER
C152 CC   0001              LDD    #$01      START TEST AT TRACK 0, SECTOR 1
C155 FD   C266              STD    CURSEC

                    * CREATE TEST PATTERN IN BUFFER 1

C158 8E   C268      TEST    LDX    #BUFF1    POINT AT START OF BUFFER 1
C15B 5F                     CLRB             COUNTER = 256
C15C B6   C261              LDA    TSTPAT    SET UP CHECKER BOARD
C15F A7   80        SETUP   STA    0,X+      STORE A BYTE
C161 43                     COMA             ALTERNATE 1's AND 0's
C162 5A                     DECB             COUNT IT
C163 26   FA                BNE    SETUP

                    * WRITE DATA TO DISK

C165 8E   C268      WRTDAT  LDX    #BUFF1    POINT AT DATA TO BE WRITTEN
C168 FC   C266              LDD    CURSEC    GET TRACK AND SECTOR BEING TESTED
C16B BD   DE03              JSR    WRITE     USE FLEX DRIVERS TWO WRITE DATA

                    * READ IT BACK

C16E 8E   C368              LDX    #BUFF2    POINT AT BUFFER 2
C171 FC   C266              LDD    CURSEC    GET TRACK AND SECTOR BEING TESTED
C174 BD   DE00              JSR    READ      USE FLEX DRIVERS TO READ DATA

                    * COMPARE DATA WRITTEN OUT TO DATA READ BACK

C177 8E   C268              LDX    #BUFF1
C17A 108E C368              LDY    #BUFF2
C17E 5F                     CLRB             SET COUNT = 256
C17F A6   80        COMPAR  LDA    0,X+
C181 A1   A0                CMPA   0,Y+
C183 26   39                BNE    DSKERR
C185 5A                     DECB
C186 26   F7                BNE    COMPAR
C188 FC   C266      DONYET  LDD    CURSEC    GET SECTOR JUST TESTED
C18B 1083 C264              CMPD   TRKMAX    AT END OF DISK?
C18F 26   16                BNE    CONTIN    NO, KEEP GOING
C191 73   C261              COM    TSTPAT    USE INVERTED PATTERN ON NEXT PASS
C194 B6   C260              LDA    PASCNT    GET PASS COUNTER
C197 4C                     INCA
C198 B7   C260              STA    PASCNT
C19B 91   00                CMPA   PASSES    DESIRED NUMBER OF PASSES COMPLETE?
C19D 27   16                BEQ    DONMSG
C19F CC   0001              LDD    #$01      START A NEW PASS
C1A2 FD   C266              STD    CURSEC
C1A5 20   B1                BRA    TEST
C1A7 F1   C263      CONTIN  CMPB   SECTRK    LAST SECTOR ON A TRACK?
C1AA 25   02                BLO    NXTSEC    NO, DO NEXT SECTOR
C1AC 4C                     INCA             YES, GO TO NEXT TRACK
C1AD 5F                     CLRB             AND CLEAR SECTOR
C1AE 5C             NXTSEC  INCB             NEXT SECTOR
C1AF FD   C266              STD    CURSEC
>C1B2 16  FFB0              LBRA   WRTDAT

C1B5 8E   C1FD      DONMSG  LDX    #MSG1
C1B8 BD   CD1E              JSR    PSTRNG
C1BB 7E   CD03              JMP    WARMS

C1BE 8E   C20B      DSKERR  LDX    #MSG2     ANNOUNCE ERROR
C1C1 BD   CD1E              JSR    PSTRNG
C1C4 8E   C217              LDX    #MSG21    PRINT TRACK NUMBER
C1C7 BD   CD1E              JSR    PSTRNG
C1CA 8E   C266              LDX    #CURSEC   GET TRACK WITH BAD SECTOR
```

```
C1CD BD    CD3C             JSR    OUTHEX
C1D0 8E    C223             LDX    #MSG22    PRINT SECTOR NUMBER
C1D3 BD    CD1E             JSR    PSTRNG
C1D6 BE    C267             LDX    CURSEC+1 GET BAD SECTOR NUMBER
C1D9 BD    CD3C             JSR    OUTHEX
C1DC BD    CD24             JSR    PCRLF
C1DF 16    FFA6             LBRA   DONXET    CONTINUE TEST WITH NEXT SECTOR

C1E2 8E    C22F     MENU    LDX    #MSG3
C1E5 BD    CD1E             JSR    PSTRNG
C1E8 BD    CD15             JSR    GETCHR
C1EB 80    30               SUBA   #$30      CONVERT ASCII TO BINARY
C1ED 81    02               CMPA   #2
C1EF 2E    03               BGT    ERROR     NO SUCH DRIVE
C1F1 16    FF28             LBRA   SELECT

C1F4 8E    C24B     ERROR   LDX    #MSG4
C1F7 BD    CD1E             JSR    PSTRNG
C1FA 7E    CD03             JMP    WARMS

C1FD 54 45 53 54    MSG1    FCC    'TEST COMPLETE',4
C201 20 43 4F 4D
C205 50 4C 45 54
C209 45 04

C20B 44 49 53 4B    MSG2    FCC    'DISK ERROR!',4
C20F 20 45 52 52
C213 4F 52 21 04
C217 20 20 54 52    MSG21   FCC    '  TRACK = ',4
C21B 41 43 4B 20
C21F 20 3D 20 04
C223 20 20 53 45    MSG22   FCC    '  SECTOR = ',4
C227 43 54 4F 52
C22B 20 3D 20 04
C22F 44 52 49 56    MSG3    FCC    'DRIVE NUMBER TO BE TESTED? ',4
C233 45 20 4E 55
C237 4D 42 45 52
C23B 20 54 4F 20
C23F 42 45 20 54
C243 45 53 54 45
C247 44 3F 20 04

C24B 49 4E 56 41    MSG4    FCC    'INVALID DRIVE NUMBER'
C24F 4C 49 44 20
C253 44 52 49 56
C257 45 20 4E 55
C25B 4D 42 45 52
C25F 04                     FCC    4

                    * VARIABLE STORAGE

C260                PASCNT  RMB    1
C261                TSTPAT  RMB    1          TEST PATTERN FOR THIS PASS

C262                DRIVE   RMB    1          DRIVE TO BE TESTED
C263                SECTRK  RMB    1          NUMBER OF SECTORS PER TRACK
C264                TRKMAX  RMB    1          LARGEST TRACK NUMBER ON DISK
C265                SECMAX  RMB    1          LAST DATA SECTOR ON LAST TRACK
C266                CURSEC  RMB    2          TRACK AND SECTOR BEING TESTED

C268                BUFF1   RMB    256        DATA WRITTEN TO DISK
C368                BUFF2   RMB    256        DATA READ FROM DISK

                            END    $C100

0 ERROR(S) DETECTED

SYMBOL TABLE:

BUFF1  C268   BUFF2  C368   COMPAR C17F   CONTIN C1A7   CURSEC C266
DOMSG  C185   DONXET C188   DRIVE  C262   DRVSEL DE0C   DSKERR C1BE
ERROR  C1F4   GETCHR CD15   MENU   C1E2   MSG1   C1FD   MSG2   C20B
MSG21  C217   MSG22  C223   MSG3   C22F   MSG4   C24B   NXTCH  CD27
NXTSEC C1AE   OUTHEX CD3C   PASCNT C260   PASSES 0000   PCRLF  CD24
PSTRNG CD1E   READ   DE00   SECMAX C265   SECTRK C263   SELECT C11C
SETUP  C15F   START  C103   TEST   C150   TRKMAX C264   TSTPAT C261
TTYEOL CC03   VN     C102   WARMS  CD03   WRITE  DE03   WRTDAT C165
```

24 Coldstream Dr.
Munster, Ontario
Canada   K0A 3P0
November 14, 1983

Mr. Don Williams
68 Micro Journal
Box 849
Hixson, TN   37343

Dear Don:

Have just finished typing in and testing the 'New Copy Program' in the November issue. Congratulations on providing a great utility that fills a real hole in FLEX support software.

There is, however, one obvious 'bug' in the program that should be corrected - **make sure that all registers are initialized properly.** In the listing, the DP register is assumed to contain a value of '00'. This is not necessarily the case.

I get the impression from the articles and letters in '68' Micro Journal that most users are only vaguely aware of the organization and structure of the handlers used in FLEX.

You will notice tnet there are no restrictions placed on the final condition of the DP register for any routine, not even WARMS. Therefore, every program should intialize all registers before they are used.

Sincerely,

Wm. Jackson

To

'68' MICRO JOURNAL
P O Box 849
Hixson, TN 37343
USA

Dear Don!

Thanks for the no 1 68 computer magazine!

I'm the secretary of a Swedish computer club, Privatdatorklubben PD68, which is dedicated to the 68-family of microcomputers.

We hope that you would like to inform your readers about our existence. This could be of mutual interest since many of our club members read your magazine, but it has come to our knowledge that many of your Swedish readers are not members of our club. And a few of our members have contributed to your magazine over the years.

The Privatdatorklubben PD68 consists of more than 350 members. About 30 of the members are Scientific institutions and Companies the rest are individual members. We also have members in both Denmark and Norway. We are the only 68 dedicated club in Sweden.

The club members of the Stockholm area meet the last thursday of the month, except December, June and July. We have a quarterly club magazine called MPU-laren, written in Swedish. Our club-computer which members may contact via 300 baud modem is frequently used. We also support members who design hardware for non-profit purposes. The next project we plan to support is a computer based on VME-bus and the 68000'

The equipment used by the members differs much but is mostly Flex2/9 based. Some of us also have OS-9 based systems.

The costs of a membership is 100 SKR a year, that's about USD 12.50.

Yours truly

Ingemar Skarpås

Ingemar Skarpås
secretary of pd68

PD68
Box 1098
S-122 21 ENSKEDE 1
Sweden

ACORN™
COMPUTER SYSTEMS SS-50C

October 28, 1983

Dear Don,

Here are a couple FLEX utilities that we have written for our "PRAM-TRANS(tm)" system. Because our system has FLEX and all its utilities in PROM, we had no need for auto STARTUP. STARTUP is still a powerful feature, so we created "RUN.CMD" to call it. We have been using it for years on business programs. Just type "RUN" instead of "XBASIC PROGRAM". You don't even have to remember the name or spelling of the program.

R.CMD is a lazy man's "repeat command line" with a prompt. It works great for printing a catalog of your disks. Just type +++R,P,CAT. When the first disk is completed, change disks and answer the prompt "REPEAT (Y-N)?" with a 'Y' as many times as needed. When done, answer 'N' and return to FLEX.

These programs follow Leo E. Taylor's concept of writing FLEX utilities that can be assembled for 6800 or 6809 CPUs. Usually this only requires changing one FLEX equate. In this way, we do not split FLEX users into two factions.

Yours truly,
Merle Giesfeldt
Merle Giesfeldt

```
                  ****************************************************
                  *   RUN.CMD        EXEC, STARTUP.TXT FILE           *
                  *   ------------------------------------------     *
                  *   +++RUN 1          STARTUP.TXT on disk 1         *
                  *   +++RUN                  Defalt to disk 0        *
                  *                                                   *
                  *   ACORN COMPUTER SYSTEMS    A.C.S.Inc.            *
                  *   By Merle Giesfeldt       Oct 24, 1983           *
                  ****************************************************
                  *
                  *   FLEX EQUATES
                  *
          C000    FLEX     EQU     $C000       ($A000 for 6800 FLEX)

          C027    NXTCHR   EQU     FLEX+$D27
          CD1E    PSTRNG   EQU     FLEX+$D1E
          C080    BUFFER   EQU     FLEX+$80
          CC14    BUFPNT   EQU     FLEX+$C14
          C006    RENTER   EQU     FLEX+$006

C100                       ORG     FLEX+$100
                  *----------------------------------*

C100 20    05     RUN      BRA     RUN1
C102 01                    FCB     1             Version number

                  *  TEMPORARY DATA
C103              BTEMP    RMB     2
C105              STEMP    RMB     2

C107 BD    CD27   RUN1     JSR     NXTCHR        Get next character
C10A 25    06              BCS     RUN2          Branch if no char.
C10C 84    0F              ANDA    #$0F          Remove ASCII
C10E 81    03              CMPA    #03           Compare to max. drive
C110 23    01              BLO     RUN3          Branch if OK
C112 4F           RUN2     CLRA                  Load drive "0"
C113 8A    30     RUN3     ORA     #$30          Change to ASCII
C115 B7    C14F            STAA    DRVNUM        Save drive number
C118 8E    C146            LDX     #STARST       "EXEC, 0.STARTUP.TXT"
C11B BD    CD1E            JSR     PSTRNG
C11E 8E    C080            LDX     #BUFFER       Load buffer address
C121 BF    CC14            STX     BUFPNT        Save in buffer pointer
C124 BF    C103            STX     BTEMP
C127 C6    14              LDAB    #20           Load string count
C129 8E    C149            LDX     #BUFFST       Load string address
C12C A6    84     RLOOP    LDAA    0,X           Get char. from string
C12E 30    01              INX
C130 BF    C105            STX     STEMP
C133 BE    C103            LDX     BTEMP
C136 A7    84              STAA    0,X           Store char. in buffer
C138 30    01              INX
C13A BF    C103            STX     BTEMP


C13D BE    C105            LDX     STEMP
C140 5A                    DECB                  Decrement count
C141 26    E9              BNE     RLOOP
C143 7E    C006            JMP     RENTER        Go do command

C146 0D 0A 0A     STARST   FCB     $D,$A,$A
C149 45 58 45 43  BUFFST   FCC     /EXEC, /
C14F 30           DRVNUM   FCB     $30           ASCII "0"
C150 2E 53 54 41           FCC     /.STARTUP.TXT/
C15C 0D 0A 04              FCB     $D,$A,4

                           END     RUN

0 ERROR(S) DETECTED
```

```
                  ****************************************************
                  *   R.CMD        REPEAT COMMAND LINE               *
                  *   ---------------------------------------        *
                  *   Uses upper half of input buffer.               *
                  *   Leaves room for 64 byte CMD string.            *
                  *   Prompts "REPEAT (Y-N)?"                        *
                  *                                                  *
                  *  +++R,P,CAT      Change dist and repeat          *
                  *  +++R,NEWDISK 1:COPY 0,1 .CMD .SYS .LOW          *
                  *                                                  *
                  *  ACORN COMPUTER SYSTEMS - A.C.S. Inc.            *
                  *    By Merle Giesfeldt   Oct. 24, 1983            *
                  ****************************************************
                  * FLEX - Trademark of T.S.C. Inc.

                  *     FLEX EQUATES

          C000    FLEX     EQU     $C000       ($A000 for 6800 FLEX)

          CD03    WARMST   EQU     FLEX+$D03
          CD4B    DOCMND   EQU     FLEX+$D4B
          CC14    BUFPNT   EQU     FLEX+$C14
          CD1E    PSTRNG   EQU     FLEX+$D1E
          CD15    GETCHR   EQU     FLEX+$D15

C0C2                       ORG     FLEX+$C2
                  *-----------------------------*

C0C2 20    03     REPEAT   BRA     REPT
C0C4 01                    FCB     1             Version number

                  *     TEMPORARY DATA
C0C5 0000         BTEMP    FDB     0000

C0C7 BE    CC14   REPT     LDX     BUFPNT        Get buffer pointer
C0CA BF    C0C5            STX     BTEMP         Save pointer
C0CD BD    CD4B   REP2     JSR     DOCMND        Do command
C0D0 26    1B              BNE     REP4          If ERROR exit
C0D2 BE    C0C5            LDX     BTEMP         Reset buffer pointer
C0D5 BF    CC14            STX     BUFPNT
C0D8 8E    C0F0   REP3     LDX     #RDYST        "REPEAT ?"
C0DB BD    CD1E            JSR     PSTRNG
C0DE BD    CD15            JSR     GETCHR        Get Y or N
C0E1 84    5F              ANDA    #$5F          Change to upper case
C0E3 81    4E              CMPA    #'N           (NO)
C0E5 27    06              BEQ     REP4          Then exit
C0E7 81    59              CMPA    #'Y           (YES)
C0E9 26    ED              BNE     REP3          Not "Y", ask again
C0EB 20    E0              BRA     REP2          Do command again

C0ED 7E    CD03   REP4     JMP     WARMST        Go to FLEX

C0F0 0A 0D        RDYST    FCB     $A,$D


C0F2 52 45 50 45           FCC     /REPEAT (Y-N)?/
C0FF 04                    FCB     4

                           END     REPEAT

0 ERROR(S) DETECTED
```

'68' Micro Journal
Attn: Don Williams
5900 Cassandra Drive
P.O. Box 849
Hixon, Tn 37343

10/14/83

Dear Mr. Williams

I have been a subscriber to your magazine for nearly two years, and I enjoy it very much. However, I see much more print devoted to software languages and comparisions than to information for hobbyists. This is a change from the earlier issues of your magazine. The reason I wrote is to make available to your readers some programs and patches I have developed over the past year. The first is a patch to DYNAMITEtm Computer Systems Center which overcomes a very frustrating shortcoming of an excellent program, the inability to input lowercase commands or addresses. The patch is called DYNP.TXT. The second is a paginating print driver with printer control strings invoked by arguements on the command line, this program is called p.txt. If anyone isinterested in a copy of these files on a floppy, I will be happy to copy them onto either a 5 1/4 or 8 inch single density formatted floppy. If anyone would like to exchange programs I will pay return postage, otherwise please provide return postage. Please continue to provide useful information to the 6800 users.

Yours truly,

Ron Roetzer
RFD Beaman Rd.
Princeton Ma
01541
(617)464-5561

```
* dynp
* patch to dynamite to allow lowercase
* in the command line
* first relocate by $100
* relocate dynamite dynx 100
* then append this patch to dynx
* append dynx.Cmd dynp.Bin dyn.Cmd

nxtch   equ     $cd27
sysfcb  equ     $c840
xferfg  equ     $cc1d
xferad  equ     $cc1e
errtyp  equ     $cc20
fms     equ     $d406

        org     $01b8
        jsr     unxtch
        org     $0289
        jsr     unxtch
        org     $0409
        jsr     unxtch
        org     $0422
        jsr     unxtch
        org     $04e0
        jsr     unxtch
        org     $05bd
        jsr     unxtch
        org     $05ff
        jsr     unxtch
        org     $07fd
        jsr     unxtch
        org     $081f
        jsr     unxtch
        org     $0836
        jsr     unxtch
        org     $03c1
        jsr     ughex
        org     $04f1
        jsr     ughex
        org     $0556
        jsr     ughex
        org     $0839
        jsr     ughex

        org     0000
unxtch  jsr     nxtch
        psh     cc
        bsr     upper
        pul     cc
        rts
upper   cmpa    #'a
        blt     rtn
        anda    #$5f
rtn     rts

flg1    fcb     0
flg2    fcb     0

ughex    clra
         clrb
         std     flg1
loop     jsr     nxtch
         bcs     carry
         bsr     hex1
         bcs     hex2
         pshs    b
         ldb     #4
loop1    asl     flg2
         rol     flg1
         decb
         bne     loop1
         puls    b
         adda    flg2
         sta     flg2
         incb
         bra     loop
hex2     jsr     nxtch
         bcc     hex2
         rts
carry    ldx     flg1
         andcc   #$fe
         rts
```

```
hex1     bsr     upper
         suba    #$47
         bpl     hex3
         adda    #6
         bpl     hex4
         adda    #7
         bpl     hex3
hex4     adda    #$0a
         bmi     hex3
         andcc   #$fe
         rts
hex3     orcc    #1
         rts

*P
*PARALLEL PRINTER DRIVER
*PRINTS 56 LINES PER PAGE THEN FORMFEED
*SUPPLIES CONTROL STRINGS FOR C-ITOH 8510 PRINTER
*CALLED P +AE LIST FILE
*A       TEN PITCH
*B       TWELVE PITCH
*C       PROPORTIONAL
*D       SIXTEEN PITCH
*E       BOLD (DOUBLE STRIKE)
*F       DOUBLE WIDTH
*G       SET LEFT MARGIN IN 8 CHARACTERS FOR LETTERS
*        OR THREE HOLE PUNCH
*H       NO AUTO FF
*LAST REVISED 24-FEB-83
*RON ROETZER

DR       EQU     0
DDR      EQU     0
CR       EQU     1
BUFPNT   EQU     $CC14
NXTCH    EQU     $CD27

         ORG     $C300
         FDB     ENDS-POPEN DRIVER LEGNTH

*ENTRY VECTORS
POPEN    LBRA    OPEN
PQUIT    LBRA    CLOSE
PCHAR    LBRA    PUTX
PCHEK    LBRA    CHECK
PIA      FDB     $E01C
SIDE     FCB     0
         FCB     0
PFLAG    FCB     $FF
TEMPS    FDB     0000
LINCNT   FCB     0
FFLAG    FCB     0

*CODES FOR TEN PITCH
TEN      LDA     #27
         LBSR     PUT
         LDA     #'N
         LBSR     PUT
         LBRA    OPTLOP

*CODES FOR TWELVE PITCH
TWELVE   LDA     #27
         LBSR     PUT
         LDA     #'E
         LBSR     PUT
         LBRA    OPTLOP

*CODES FOR PROPORTIONAL PITCH
PROP     LDA     #27
         LBSR     PUT
         LDA     #'P
         LBSR     PUT
         LBRA    OPTLOP

*CODES FOR SIXTEEN PITCH
SIXTEN   LDA     #27
         LBSR     PUT
         LDA     #'Q
         LBSR     PUT
         LBRA    OPTLOP
```

```
*CODES FOR DOUBLE STRIKE
BOLD     LDA      #27
         LBSR      PUT
         LDA      #'!
         LBSR      PUT
         LBRA     OPTLOP

*CODES FOR DOUBLE WIDTH
WIDE     LDA      #14
         LBSR      PUT
         LBRA     OPTLOP

*CODES FOR INDENT LEFT MARGIN 8 CHARACTERS
DRAFT    LDA      #27
         LBSR      PUT
         LDA      #'L
         LBSR      PUT
         LDA      #'0
         LBSR      PUT
         LDA      #'0
         LBSR      PUT
         LDA      #'8
         LBSR      PUT
         LBRA     OPTLOP

*NO AUTO FF
NOFF     INC      FFLAG,PCR
         LBRA     OPTLOP


OPEN     LDX      PIA,PCR
         LDA      #$3A
         STA      CR,X
         LDA      #$FF
         STA      DDR,X
         LDA      #$3E
         STA      CR,X
         LDA      #$0D       CARRIAGE RETURN
         LBSR      PUT
         LDA      #$0C       FORM FEED
         LBSR      PUT
         LDA      #27        RESET LEFT MARGIN
         LBSR      PUT
         LDA      #'L
         LBSR      PUT
         LDA      #'0
         LBSR      PUT
         LDA      #'0
         LBSR      PUT
         LDA      #'0
         LBSR      PUT
         LDA      #27        SET TEN PITCH
         LBSR      PUT
         LDA      #'N
         LBSR      PUT
         LDA      #27        CLEAR DOUBLE STRIKE
         LBSR      PUT
         LDA      #34
         LBSR      PUT
         LDA      #15        CLEAR DOUBLE WIDTH
         LBSR      PUT

*PARSE THE INPUT LINE FOR +
*AND CHECK FOR A THRU H

SAVBUF   LDX      BUFPNT
         STX      TEMP5,PCR  POSITION
SCAN     JSR      NXTCH SCAN LINE FOR LETTERS
         CMPA     #'+
         BEQ      SAVBUF IGNORE PLUS
         CMPA     #$60       CHECK FOR LOWER CASE
         BLE      CTEST
         ANDA     #$5F       AND COVVERT TO UPPER CASE
CTEST    CMPA     #'A
         BLO      SCAN2
         CMPA     #'H
         BLS      SCAN
SCAN2    LDX      TEMP5,PCR
         STX      BUFPNT      RESET POINTER
         CMPA     #$20        SPACE FOLLOWS OPTIONS
         BNE      TRY1
```

```
         BRA      OPTLOP
TRY1     CMPA     #',
         BNE      NOOPT
OPTLOP   JSR      NXTCH       GET OPTION CHAR
         BCS      ENDOPT      NON-ALPHANUMERIC
         CMPA     #$60        SET UP PRINTER
         BLE      A
         ANDA     #$5F
A        CMPA     #'A
         BNE      B
         LBRA     TEN
B        CMPA     #'B
         BNE      C
         LBRA     TWELVE
C        CMPA     #'C
         BNE      D
         LBRA     SIXTEN
D        CMPA     #'D
         BNE      E
         LBRA     PROP
E        CMPA     #'E
         BNE      F
         LBRA     BOLD
F        CMPA     #'F
         BNE      G
         LBRA     WIDE
G        CMPA     #'G
         BNE      H
         LBRA     DRAFT
H        CMPA     #'H
         BNE      ENDOPT
         LBRA     NOFF
         BRA      OPTLOP

ENDOPT   LDX      BUFPNT
         LDAA     0,X
         CMPA     #$20        SPACE?
         BEQ      SKPSEP
         CMPA     #',         COMMA?
         BNE      NOOPT
SKPSEP   JSR      NXTCH       PASS OVER SEPARATOR
NOOPT    CLR      LINCNT,PCR
         RTS

CLOSE    LDA      #$0D

PUTX     TST      FFLAG,PCR
         BNE      PUT
         CMPA     #$0D        COUNT NUMBER OF LINES
         BNE      PUTY        ADD FORMFEED ON THE 56TH
         INC      LINCNT,PCR
         BSR      PUT
         LDA      LINCNT,PCR
         CMPA     #56
         BNE      PUTRTN
         CLR      LINCNT,PCR
         LDA      #$0C        IF FORMFEED CLEAR LINCNT
         BRA      PUT
PUTRTN   RTS
PUTY     CMPA     #$0C IF NO CR OF FF PRINT NORMALLY
         BNE      PUT
         CLR      LINCNT,PCR

PUT      LBSR     CHECK
         BPL      PUT
         PSHS     X
         CLR      PFLAG,PCR
         LDX      PIA,PCR
         STA      DR,X
         LDA      #$36
         STA      CR,X
         LDA      #$3E
         STA      CR,X
         PULS     X,PC

CHECK    PSHS     X
         TST      PFLAG,PCR
         BMI      CHEXIT
         LDX      PIA,PCR
```

```
        TST     CR,X
        BPL     CHEXIT
        TST     OR,X
        COM     PFLAG,PCR
CHEXIT  PULS    X,PC
ENDS    EQU     *
        END
```

```
10 REM**      PRECOMP.BAS
11 REM**            CONVERTS 'ERL' AND 'CHAIN' LABELS TO EPC LINE NUMBERS.
20 ON ERROR GOTO 9900
25 PRINT CHR$(16)+CHR$(22) : PRINT
30 DIM FL$(100)
32  DIM LB$(200,2)
40 LC=1 : LI=60 : LI=0
50 PRINT : PRINT
52 PRINT"ENTER INPUT DEVICE  , #0 (KEYBOARD) , OR  FILENAME ";:INPUT LINE I0$
54 IF I0$="#0" THEN 65  ELSE  GOTO 2000
65 FOR I=1 TO 100
70 PRINT CHR$(16)+CHR$(22)
80 PRINT : PRINT
90 PRINT"ENTER FILE NAME(S) OR  EXIT ";:INPUTLINE FL$(I)
92 IF  FL$(I)="EXIT" THEN I=I-1 : GOTO 100
93 IF LEN(FL$(I))=0 THEN I=I-1 : GOTO 100
95 NEXT I
100 J=0
110 J=J+1 : IF J > I THEN 9970
112 TM=0 : LN=0 : LP=0 : LE=0 : XP=0 : LL=0 \ CLOSE 0 \ EP=0
113 E4=INSTR(1,FL$(J),"+")
114 IF E4=0 OR E4=LEN(FL$(J)) THEN 120
115 LN=INSTR(E4,FL$(J),"L")
116 LP=INSTR(E4,FL$(J),"P")
117 LE=INSTR(E4,FL$(J),"E")
118 XP=INSTR(E4,FL$(J),"X")
119 LL=INSTR(E4,FL$(J),"T")
120 FOR E1=1 TO LEN(FL$(J))
121 IF INSTR(1,",  ",MID$(FL$(J),E1,1)) THEN 124
122 NEXT E1
124 IF INSTR(1,"0123456789",LEFT$(FL$(J),1))=0 THEN E2=INSTR(1,FL$(J),".")
        ELSE E2=INSTR(3,FL$(J),".")
125 IF E1>LEN(FL$(J)) THEN FI$=FL$(J) ELSE FI$=LEFT$(FL$(J),E1-1)
126 FO$=LEFT$(FI$,E2)+"COM"
130 FOR T=E1 TO LEN(FL$(J))
131 IF INSTR(1,"  ",MID$(FL$(J),T,1))=0 THEN GOTO 140
132 NEXT T
134 GOTO 170
140 FOR E3=T TO LEN(FL$(J))
141 IF INSTR(1,",  ",MID$(FL$(J),E3,1))<>0 THEN 150
142 NEXT E3
143 IF E3=T THEN GOTO 170
150 IF E3-T > 1 THEN FO$=MID$(FL$(J),T,E3-T)
170 OPEN OLD FI$ AS 2
180 OPEN NEW FO$ AS 3
184 IF LP=0 THEN 200
186 EXEC,"ASN,W=A"
188 OPEN "PRINT" AS 0
190 EXEC,"ASN,W=0"
200 INPUTLINE#2,A$
230 LI=LI+1
240 IF LEN(A$) > 70 THEN LI=LI+1
250 IF LI > 56 THEN GOSUB 1000
300 IF TF=0 THEN C=C+1 : IF LN=0 THEN PRINT#0%,C;TAB(5);
310 IF RIGHT$(A$,1)="\" THEN TF=1
320 IF LN=0 THEN PRINT#0%,TAB(5);LEFT$(A$,74)
340 IF LEN(A$) > 74 AND LN=0 THEN PRINT#0%,TAB(10);MID$(A$,75)
360 IF RIGHT$(A$,1)<>"\" THEN TF=0
400 REM**
410 IF LEFT$(A$,1)="@" AND INSTR(1,A$,"L10")<>0 THEN GOTO 6000
420 IF LEFT$(A$,1)<>"@" THEN 600
430 E=INSTR(1,A$," ")
432 IF E=0 THEN E=LEN(A$)+1
440 LB=LB+1
450 LB$(LB,2)=LEFT$(A$,E-1)
460 LB$(LB,1)=STR$(C)
600 GOTO 200
1000 REM
1004 IF LN <> 0 THEN 1090
1005 PI=PI+1
1010 PRINT #0%,CHR$(12)
1030 LI=1
1040 PRINT#0%," "
1050 PRINT#0%,"   L I S T I N G   F O R    ";FI$
1060 PRINT#0%,"              ";DATE$;"          PAGE ";PI
1070 PRINT#0%," ";PRINT#0%," "
1090 RETURN
2000 REM**
2020 OPEN OLD I0$ AS 1
2030 FOR I=1 TO 100
2040 INPUT LINE #1,FL$(I)
2050 NEXT I
2100 I=I-1
2120 CLOSE 1
2130 GOTO 100
4000 REM**
4002 TM=TM+1
4004 IF LN=0 THEN PRINT#0%
4005 IF LE=0 THEN CLOSE 0 : GOTO 4020
4006 IF LP > 0 THEN 4020
4010 EXEC,"ASN,W=A"
4012 OPEN "PRINT" AS 0
4014 EXEC,"ASN,W=0"
4020 OPEN OLD FI$ AS 2
4030 TF=0 : CX=0 : CE=0
4040 INPUTLINE #2,A$
4042 CE=CE+1
4044 IF TF=0 THEN CX=CX+1 \ TL%=0
4046 IF RIGHT$(A$,1)="\" THEN TF=1
4048 IF RIGHT$(A$,1)<>"\" THEN TF=0
4050 IF LL=0 THEN 4100
4060 QU=0 : FO=0 : PA%=0
4062 FOR T=1 TO LEN(A$)
4064 TA$=MID$(A$,T,1)
4066 IF TA$='"' OR TA$="'" THEN QU=1-QU
4068 IF QU=1 THEN 4074
4070 IF TA$<>" " THEN FO=0 \ GOTO 4074
4072 IF TA$=" " AND FO=1 THEN 4076  ELSE  FO=1
4074 PA%=PA%+1
4076 NEXT T
4080 TL%=TL%+PA%
4082 IF TF=0 AND TL%>255 THEN MS$="LONG LINE,LENGTH = "+STR$(TL%) \GOTO 4500
4100 E=INSTR(1,A$,"ERL")
4110 IF E > 0 THEN 4200
4120 E=INSTR(1,A$,"CHAIN")
4130 IF E=0 THEN 4600
4180 PRINT #3,A$
4190 GOTO 4040
4200 REM**
4210 FOR E1=E+3 TO LEN(A$)
4220 IF INSTR(1,"< >=",MID$(A$,E1,1))=0 THEN GOTO 4260
4230 NEXT E1
4232 MS$="ERL SYMBOL ERROR"
4240 GOTO 4500
4260 FOR E2=E1 TO LEN(A$)
4270 IF MID$(A$,E2,1)=" " THEN GOTO 4310
4280 NEXT E2
4290 IF E2>LEN(A$) THEN GOTO 4310
4292 MS$="ERL LABEL ERROR"
4300 GOTO 4500
4310 A1$=MID$(A$,E1,E2-E1+1)
4340 FOR L=1 TO LB
4350 IF LB$(L,2)=A1$ THEN A2$=LB$(L,1) : GOTO 4380
4360 NEXT L
4370 MS$="LABEL NOT FOUND"
4372 GOTO 4500
4380 PRINT#3,LEFT$(A$,E1-1)+A2$+MID$(A$,E2)
4390 GOTO 4040
4500 REM**
4510 PRINT #0%,  TAB(5);A$
4520 PRINT #0%,  "** ERROR ** ";CE;CX;" ";MS$
4540 EP=EP+1
4590 GOTO 4180
4600 REM**
4620 E1=INSTR(E,A$,'"')
4622 IF E1=0 THEN 4180
4630 E2=INSTR(E1+1,A$,'"')
4632 IF E2=0 THEN 4180
4640 IF LEN(A$)<E2 THEN 4180
4650 FOR T=E2+1 TO LEN(A$)
4660 IF MID$(A$,T,1)<>" " THEN 4700
4670 NEXT T
4680 MS$="CHAIN LABEL ERROR"
4690 GOTO 4500
4700 A1$=MID$(A$,T)
4710 CF$=MID$(A$,E1+1,E2-E1-1)
4720 IF MID$(FI$,2,1)="." THEN CF$=LEFT$(FI$,2)+CF$
4730 OPEN OLD CF$+RIGHT$(FI$,4) AS 4
4732 C1=0 : TF=0
4740 INPUTLINE #4,FA$
4760 IF TF=0 THEN C1=C1+1
4770 IF RIGHT$(FA$,1)="\" THEN TF=1
4780 IF RIGHT$(FA$,1)<>"\" THEN TF=0
4800 IF LEFT$(FA$,LEN(A1$))=A1$ THEN GOTO 4840
4810 GOTO 4740
4840 PRINT#3,LEFT$(A$,T-1)+STR$(C1)
4900 CLOSE 4
4890 GOTO 4040
6000 REM**
6010 E1=INSTR(1,A$,"L10")
6020 FOR T=E1+3 TO LEN(A$)
6030 IF MID$(A$,T,1)<>" " THEN GOTO 6060
6040 NEXT T
6050 GOTO 200
6060 OPEN OLD MID$(A$,T) AS 4
6200 INPUTLINE #4,AF$
6220 LI=LI+1
6240 IF LEN(AF$) > 74 THEN LI=LI+1
6250 IF LI > 56 THEN GOSUB 1000
6280 IF TF=0 THEN C=C+1 : IF LN=0 THEN PRINT#0%,C;TAB(5);
6300 IF RIGHT$(AF$,1)="\" THEN TF=1
6320 IF LN=0 THEN PRINT#0%,TAB(5);LEFT$(AF$,74)
6340 IF LEN(AF$) > 74 AND LN=0 THEN PRINT#0%,TAB(10);MID$(AF$,75)
6360 IF RIGHT$(AF$,1)<>"\" THEN TF=0
6400 REM**
6410 IF LEFT$(AF$,1)<>"@"  THEN GOTO 6200
6420 IF LEFT$(AF$,1)<>" " THEN 6200
6430 E=INSTR(1,AF$," ")
6432 IF E=0 THEN E=LEN(AF$)+1
6440 LB=LB+1
6450 LB$(LB,2)=LEFT$(AF$,E-1)
6460 LB$(LB,1)=STR$(C)
6600 GOTO 6200
8000 CLOSE 2
8020 GOTO 170
8100 CLOSE 1 : GOTO 30
9900 REM
9901 PRINT:PRINT
9903 IF ERR=8 AND ERL=200 THEN RESUME 9950
9904 IF ERR=8 AND ERL=2020 THEN PRINT FI$;" NOT FOUND" : RESUME 110
9905 IF ERR=21 AND ERL=170 THEN PRINT FI$;" ILLEGAL FILE" : RESUME 110
9906 IF ERR=41 AND ERL=170 THEN RESUME 8000
9910 IF ERR=4 AND ERL=2040 THEN RESUME 50
9911 IF ERR=21 AND ERL=2020 THEN RESUME 50
9912 IF ERR=8 AND ERL=2040 THEN RESUME 2100
9913 IF ERR=41 AND ERL=2020 THEN RESUME 8100
9914 IF ERR=8 AND ERL=4040 THEN RESUME 9950
9915 IF ERR=8 AND ERL=6200 THEN CLOSE 4 \ RESUME 200
9916 IF ERR=8 AND ERL=4740 THEN CLOSE 4:MS$="CHAIN TO LABEL NOT FOUND"
        : RESUME 4500
9920 PRINT ERR;ERL:STOP
9950 REM
9960 CLOSE 3
9972 LC=1 : PI=0 : LI=60 : C=0
9975 IF TM=0 THEN 4000
9976 PRINT#0% : PRINT#0%
9977 PRINT#0%,  "NUMBER OF LABELS = ";LB;"(C), TOTAL ERRORS = ";EP
9978 CLOSE 3 : GOTO 110
9980 REM**
9981 PRINT:PRINT"FREE=";FRE(0),"LB=";LB
9999 END
```

UPPER&LOWER CASE CONV·

Dear Don:

Enclosed please find listings and disk for a program that I wrote to do upper and lower case conversions. I find that I am doing all of my programming in mixed case and this allows me to quickly convert old routines to lower case that were orginally written in upper case.

It is written in TSC precompiled, extended BASIC using some of my favorite techniques. I prefer the use of the precompiler as it makes the code much more readable with the use of extended labels and variable names.

Readers can receive complete sources by sending a FLEX9 formatted 8" disk to the above address with three dollars for handeling. (Sorry, I don't have 5" drives.)

dwight m. lanpher

BOX 472 SUMMIT ROAD
NORTHEAST HARBOR, MAINE 04662
TEL (207) 276-5350

```
+++XBASIC UPPER

UPPER/LOWER case conversion routine
(c)1983 D.M. Lanpher
version 1.3:0  November 26, 1983

         full name of file to be converted (ex. FILE.TXT)? TEST.TXT
the orignal has been renamed UPPER.TMP
do you wish to convert to uppercase or lowercase (U or L)? U
                        what is the first line to be converted? 1
   do you want the conversion through the end of the file? 20
   do you want the conversion through the end of the file? N
                         what is the last line to be converted? 20
                      what is the first column to be converted? 5
   do you want the conversion through the end of the line? N
                       what is the last column to be converted? 40

please wait --- processing information

Total of 265 lines processed
         331 characters converted
         868 characters not converted
do you wish to delete UPPER.TMP? N


+++
***********************************************************************
*                                                                     *
*                    UPPER.TXT                                        *
*                                                                     *
*           author: D.M. LANPHER                                      *
*         version$="1.3:0  November 26, 1983"
*                                                                     *
*   this program will allow the user to convert an ASCII text file    *
*   from upper to lower case or from lower to upper. The user is      *
*   also allowed to control the line range and the column range.      *
*   For instance, the first column in lines 10 through 35 could be    *
*   converted to upper case.                                          *
*   Note that the input file is renamed "UPPER.TMP" and the output    *
*   file is named the same as the original input file.  This makes    *
*   file conversions more convenient if you no longer want the        *
*   original file. The operator is given a chance to delete the       *
*   UPPER.TMP file if he wishes to do several conversions.            *
*   At the end of the run a short summary of the conversions will     *
*   be displayed.                                                     *
*                                                                     *
***********************************************************************
$TTL UPPER.TXT
$STTL (c)1983 D.M. LANPHER

*
* main routine
*
* exit is done through the error handling routine as an end
* of file error
*


main      print
          print "UPPER/LOWER case conversion routine"
          print "(c)1983 D.M. Lanpher"
          print "version ":version$
          print
          on error goto fixit
          gosub getfile                        :rem get filenames
          gosub conprm                         :rem get conversion parameters
          gosub linprm                         :rem get line parameters
          gosub colprm                         :rem get column parameters
          print
          print "please wait --- processing information"
          print

lineloop input line #1, infield$
                linecount%=linecount%+1
                if linecount%<firstline% then outfield$=infield$ \
                   :goto outline
                if linecount%>lastline%  then outfield$=infield$ \
                   :goto outline
                gosub convert                  :rem do conversion
outline         print #2, outfield$
          goto lineloop

finish   print
         print "Total of":linecount%:"lines processed"
         print "          ":convertcount:"characters converted"
         print "          ":noconvertcount:"characters not converted"
         print
         print
         gosub lowlight
         print "do you wish to delete UPPER.TMP":
         g sub highlight
         input response$
         response$=left$(response$,1)
         if response$="N" or response$="n" then goto finish2
         if response$<>"Y" and response$<>"y" then goto finish
         gosub lowlight
         print "are you sure":
         gosub highlight
         input response$
         response$=left$(response$,1)
         if response$="N" or response$="n" then goto finish2
         if response$<>"Y" and response$<>"y" then goto finish
         kill "UPPER.TMP"
         print "UPPER.TMP deleted!"

finish2  close 1,2
finish3  print
         exec,"jump,cd00"                :rem return to flex

* note:  the last line could be replaced with "end" to return to XBASIC

***********************************************************************
*
* error handling
* this also does an orderly exit for sequential files

fixit     if err=8 then resume finish
          if err=4 then print :print "file not found"\
                     :print \
                     :exec,"files" \
                     :print \
                     :resume getfile
          if err=3 then print \
                     :print "temporary file UPPER.TMP already exists"\
                     :print "please delete or rename UPPER.TMP"\
                     :resume finish3
          print "error number":err:" at line number":erl
          end

***********************************************************************
*
* get file names
*
* the following arguments are returned:
*
* infilename$    file data is to be read from


* outfilename$   file data is to be written to

getfile   gosub lowlight
          print "   full name of file to be converted (ex. FILE.TXT)":
          gosub highlight
          input infilename$
          rename infilename$,"UPPER.TMP"
          print "the original has been renamed UPPER.TMP"

          outfilename$=infilename$
          infilename$="UPPER.TMP"

          open old infilename$ as 1
          open new outfilename$ as 2
          return

***********************************************************************
*
* get conversion parameter from terminal
* the following arguments are returned:
*
* conversion$     "u" for lower to upper case conversion
*                 "l" for upper to lower case conversion

conprm    gosu  lowlight
          print "do you wish to convert to uppercase or lowercase (U or L)":
          gosub highlight
          input conversion$
          conversion$=left$(conversion$,1)
          if conversion$ = "L" then conversion$="l"
          if conversion$ = "U" then conversion$="u"
          if conversion$ <> "u" and conversion$ <> "l" then goto conprm
          return

***********************************************************************
*
* get column parameters from terminal
* the following arguments are returned:
*
* firstcol%       the first column to be converted
* lastcol%        the last column to be converted
*                 this value is set to 9999 to convert through the end of field

colprm    gosub lowlight
          print "                    what is the first column to be converted":
          gosub highlight
          input firstcol%
          if firstcol%<1 then goto colprm

colprm2   gosub lowlight
          print "   do you want the conversion through the end of the line":
          gosub highlight
          input response$
          response$=left$(response$,1)
          if response$="Y" or response$="y" then lastcol%=9999: goto colprm4

          if response$<>"N" and response$<>"n" then goto colprm2

colprm3   gosub lowlight
          print "                     what is the last column to be converted":
          gosu  highlight
          input lastcol%
colprm4   return

***********************************************************************
*
* get line parameters from terminal
* the following arguments are returned:
*
* firstline%      the first line to be converted
* lastline%       the last line to be converted
*                 this value is set to 99999 to convert through the end of file

linprm    gosub lowlight
          print "                    what is the first line to be converted":
          gosub highlight
          input firstline%
          if firstline%<1 then goto linprm

linprm2   gosub lowlight
          print "   do you want the conversion through the end of the file":
          gosub highlight
          input response$
          response$=left$(response$,1)
          if response$="Y" or response$="y" then lastline%=9999: goto linprm4
          if response$<>"N" and response$<>"n" then goto linprm2

linprm3   gosub lowlight
          print "                     what is the last line to be converted":
          gosub highlight
          input lastline%

linprm4   return

***********************************************************************
*
* this is the main conversion loop
* following arguments must be passed:
*
* infield$        the string to be converted
* firstcol%       the first column to be converted
* lastcol%        the last column to be converted
*                 set this value to 9999 to convert through the end of field
* conversion$     "u" for lower to uppec case conversion
*                 "l" for upper to lower case conversion
*
* outfield$  is returned with the conversions
*

convert   fieldlength% = len(infield$)
          outfield$=""


lineloop2 for pointer%=1 to fieldlength%

                 letter$=mid$(infield$,pointer%,1)
                 letter%=asc(letter$)
                 if pointer%<firstcol% or pointer%>lastcol% then goto otherchar
                 if conversion$="u" goto uppercase
                 if conversion$="l" goto lowercase
                 print "conversion case specified incorrectly"
                 return

lowercase              if letter%<65 or letter%>90 then go o otherchar
                       outfield$=outfield$+chr$(letter%+32)
                       convertcount=convertcount+1
                       goto convertend

uppercase              if letter%<97 or letter%>122 then goto otherchar
                       outfield$=outfield$+chr$(letter%-32)
                       convertcount=convertcount+1
                       goto convertend

otherchar              outfield$=outfield$+letter$
                       noconvertcount=noconvertcount+1

convertend next pointer%
           return

********************************************************************
********************************************************************
********************************************************************
```

```
* The following subroutines cause a SWTPC 82xx CRT to display low
* intensity and high intensity characters. They are not necessary
* for the operation of the program and could be simply replaced by
* returns or the appropriate control sequences for another terminal.
*
* example :        lowlight   return
*                  highlight  return


lowlight   print chr$(30):chr$(22):
           return

highlight  print chr$(30):chr$(8):
           return

TOTAL ERRORS = 0
```

MTP **Micro Technical Products Inc**

[illegible] N. [illegible], Suite [illegible], Mesa, AZ [illegible] (602) [illegible]

New York [illegible] Seventh Avenue, Suite [illegible], New York, NY [illegible] (212) [illegible]

'68' Micro Journal
5900 Cassandra Smith Road
Hixson, TN 37343

Would you please be so kind as to inform your readers that Micro Technical Products, Inc. is moving! Our new address will be:

Micro Technical Products, Inc.
814 W. Keating Avenue
Mesa, AZ 85202

Mr. Don Williams
'68 Micro Journal
P.O. Box 849
5900 Cassandra Smith
Hixson, TN 37343

Dear Don :

The following program has been very useful to me, maybe some of your readers will too. The program came in to being out of the need to check that my drives were o.k. speed wise. I had a problem were a file made on one drive could not be read on another drive and visa verse. This program helped me find that one of my drives was running to fast, and a quick adjustment corrected the problem. Note that this program uses timing loops set for 1 Mhz. use at 2 Mhz is o.k. as long as you are not using slow I/O. If you are using slow I/O the timing loops will have to be re-calibrated due to instructions in the I/O area. This program will only work for 5" drives and any controller using a 17XX or a 27XX controller chip. you must however change the chip register equates.

Joseph M Aulicino

Joseph M. Aulicino
2014-59th street
Bklyn, N.Y. 11204

```
                        * DISK RPM TEST
                        * Program is set up for 1mhz cpu
                        * Gentoo is RPM <drive #>

          E014          DRVREG  EQU   $E014      - Drive register
          E018          COMREG  EQU   $E018      - Command register
          CD03          WARMS   EQU   $CD03      - Flex Warm start
          CD39          OUTDEC  EQU   $CD39      - Flex Output decimal
          CD24          PCRLF   EQU   $CD24      - Flex Print Crlf
          CD42          GETHEX  EQU   $CD42      - Flex Get Hex Char.
          CD1E          PSTRNG  EQU   $CD1E      - Flex Print string

0100                            ORG   $0100

                        * Temp Storage

0100                    NUM     RMB   2
0102                    NUM1    RMB   1
0103                    TEMP    RMB   2

0105 BD   CD42          START   JSR   GETHEX     - Get drive #
0108 25   5B                    BCS   ERR
010A BF   0103                  STX   TEMP
010D F6   0104                  LDB   TEMP+1
0110 2D   53                    BLT   ERR        - Check if valid drive #
0112 C1   03                    CMPB  #3
0114 2E   4F                    BGT   ERR
0116 F7   E014                  STB   DRVREG     - Select drive
0119 8E   0187                  LDX   #MSG2      - Print start message
011C BD   CD1E                  JSR   PSTRNG
011F 86   D4                    LDA   #$D4       - 179X command for interrupt
0121 B7   E018                  STA   COMREG     - on index pulse
0124 C6   FF                    LDB   #$FF       - Loop thru program 256 times
0126 F7   0102                  STB   NUM1
0129 108E 0000          START1  LDY   #$0000     - clear count reg.
012D B6   E018                  LDA   COMREG     - Wait for start of index pulse
0130 84   02                    ANDA  #$02
0132 27   F5                    BEQ   START1
0134 31   21            COUNT   LEAY  1,Y        - count index pulse (20 us. loop)
0136 21   FC                    BRN   COUNT
0138 12                         NOP
0139 B6   E018                  LDA   COMREG
013C 84   02                    ANDA  #$02
013E 26   F4                    BNE   COUNT
0140 31   21            COUNT1  LEAY  1,Y        - count till next pulse (20 us.)
0142 21   FC                    BRN   COUNT1
0144 12                         NOP
0145 B6   E018                  LDA   COMREG
0148 84   02                    ANDA  #$02
014A 27   F4                    BEQ   COUNT1
014C 10BF 0100                  STY   NUM        - store count time
0150 8E   0100                  LDX   #NUM
0153 BD   CD39                  JSR   OUTDEC     - Print count time
0156 BD   CD24                  JSR   PCRLF
0159 F6   0102                  LDB   NUM1
015C 5A                         DECB             - decrement loop count
015D F7   0102                  STB   NUM1
0160 26   C7                    BNE   START1
0162 7E   CD03                  JMP   WARMS
0165 8E   016E          ERR     LDX   #MSG1
0168 BD   CD1E                  JSR   PSTRNG
016B 7E   CD03                  JMP   WARMS

016E 4E 6F 74 20        MSG1    FCC   'Not a valid drive number',4
0172 61 20 76 61
0176 6C 69 64 20
017A 64 72 69 76
017E 65 20 6E 75
0182 6D 62 65 72
0186 04
0187 54 68 65 20        MSG2    FCC   'The number output by the program should',$0D
,$0A
018B 6E 75 6D 62
018F 65 72 20 6F
0193 75 74 70 75
0197 74 20 62 79
019B 20 74 68 65
019F 20 70 72 6F
01A3 67 72 61 6D
01A7 20 73 68 6F
01AB 75 6C 64 0D
01AF 0A
01B0 31 30 30 30                FCC   '10000 this times 20 us. is 200 ms. this is
,$0D,$0A
01B4 30 20 74 68
01B8 69 73 20 74
01BC 69 6D 65 73
01C0 20 32 30 20
01C4 75 73 2E 20
01C8 69 73 20 32
01CC 30 30 20 6D
01D0 73 2E 20 74
01D4 68 69 73 20
01D8 69 73 20 0D
01DC 0A
01DD 74 68 65 20                FCC   'the proper time for 5 inch drives @ 1mhz cp
u.',4
01E1 70 72 6F 70
01E5 65 72 20 74
01E9 69 6D 65 20
01ED 66 6F 72 20
01F1 35 20 69 6E
01F5 63 68 20 64
01F9 72 69 76 65
01FD 73 20 40 20
0201 31 6D 68 7A
0205 20 63 70 75
0209 2E 04

                                END   START

0 ERROR(S) DETECTED

SYMBOL TABLE:

COMREG E018   COUNT  0134   COUNT1 0140   DRVREG E014   ERR    0165
GETHEX CD42   MSG1   016E   MSG2   0187   NUM    0100   NUM1   0102
OUTDEC CD39   PCRLF  CD24   PSTRNG CD1E   START  0105   START1 0129
TEMP   0103   WARMS  CD03
```

HELP

WANTED INFORMATION ON SWTPC MP-QP QUME DAISY WHEEL PRINTER INTERFACE. ANY HELP LOCATING A MP-QP BOARD OR CONSTRUCTION DETAILS FOR A SIMILAR BOARD WOULD BE APPRECIATED ALL EXPENSES WILL BE REPAID.
Ray Evans P O Box 36 Iluka N.S.W. 2460 Australia.

●●●

CLASSIFIED ADVERTISING

A/C 800 338-6800

●●●

LIKE NEW - ADDS-VIEWPOINT CRT TERMINALS - DEMO UNITS, THESE ARE LIKE NEW - WARRANTY NEVER REGISTERED - SAVE - ONLY **$478.00 EACH** - ONLY 1 LEFT. CALL ASK FOR DON, TOM or LARRY

A/C 800 338-6800

●●●

GIMIX II CPU+, 2 Mhz, with FLEX and OS-9 - Level 1, Ver. 1.2, latest as of 9/1/83. This CPU board is like new, we went to GIMIX III system. Set up and ready to run. With FLEX and OS-9. Have both on your system and just call FLEX or OS-9. Includes software and documentation. Also Time of day clock, FLEX DAT, OS-9 FPLA. Set up to require GIMIX #68 DMA disk controller. Original cost over $900.00. Special only 729.95 plus shipping (3.50). Ask for Don, Tom or Bob - 1 615 842-4601. **This will not last long!!**

●●●

TELETYPE Model 43 PRINTER - with serial (RS232) Interface, and full ASCII keyboard. **LIKE NEW** - New cost $1295.00 - ONLY **$759.00** ready to run - Call Tom - Larry - Bob, CPI 615 842-4600

CMS 9670-120D 20Mb + 1Mb Floppy, 2Mhz system. 64k static, OS9 levl ver.1.2 16 slot card-cage, switching p.s. Pascal, Basic09, C, Assembler, Debugger, Stylograph an much more. Flex + 200+ programs available. Value over $11,000. All for $6,500 O.B.O.
Friedman 604-986-2607 after 1800 Pacific time.

***

2-DSDD 8" REMEX RFD 4000A Drives in Cabinet with P/S and manual $550.00.
Joseph A Goze 312-799-6492 after 5:30

***

Complete 56K 6809 System with Gimix mainframe, DMAF-1 dual 8" disk system and Perkin-Elmer 1100 terminal. Two serial, one parallel port, MP-R EPROM Programmer and much more. FLEX-09 and loads of software. A good buy at $1,500.
John Ta vin 408-946-5700 ext. 216.

***

SS-50 2 MHz 6809 SYSTEM, H-19 CRT, GIMIX G28 CONTROLLER, PERCOM DRIVE, 2 THOMAS 48K BOARDS, TANNER 64K CARD, MP-LA, SERIAL CARD, CLOCK, F & D FMB-1 VIDEO, Z80 CARD, GOLD CONNECTORS, FLEX/OS9 SWITCHABLE, SOFTWARE & MANUALS. $1295/BEST OFFER. ALSO SWTPC 8K CARD $30, SE MICRO DDC-16 CONTROLLER $100, DMC 12" MONITOR $75.
Jim Harmon, 1509 N 10, Durant OK 405-924-4471.

***

SWTPC 6809 System, H19 CRT, 2 MPI B52 DSDD drives, Gimix G28 controller, MPS2, MPL2, MPT, 56K RAM, Software, Manuals. $1595.
Tom J Harmon, 15418 Diana Ln., Houston, TX 77062 713-480-6075.

***

FOR SALE!!
3 DM-64 HAZELWOOD MEMORY
CARDS WORK GREAT. $275.00 Ea.
(1981 not used for 2 years)
Contact Dave: 208-939-6426

***

COMPILER EVALUATION SERVICES
By: Ron Anderson

The S.E. MEDIA Division of Computer Publishing Inc., Is offering the following 'SUBSCRIBER SERVICE':

COMPILER COMPARISION AND EVALUATION REPORT

Due to the constant and rapid updating and enhanchment of numerous compilers, and the different utility, appeal, speed, level of communication, memory usage, etc., of different compilers, the following services are now being offered with periodic updates.

This service, with updates, will allow you who are wary or confused by the various claims of compiler vendors, an opportunity to review comparisons, comments, benchmarks, etc., concerning the many different compilers on the market, for the 6809 microcomputer. Thus the savings could far offset the small cost of this service.

Many have purchased compilers and then discovered that the particular compiler purchased either is not the most efficient for their purposes or does not contain features necessary for their application. Thus the added expense of purchasing additional compiler(s) or not being able to fully utilize the advantages of high level language compilers becomes too expensive.

The following COMPILERS are reviewed initially, more will be reviewed, compared and benchmarked as they become available to the autho :

PASCAL "C" GSPL WHIMISCAL PL/9

Initial Subscription - $39.95
(Includes 1 year updates)
Updates for 1 year - $14.50

S.E. MEDIA - CPI
5900 Cassandra Smith, POB 794
Hixson, TN 37343
615 842-4601

WORD PROCESSING SOFTWARE

SPELL 'N FIX finds and fixes your spelling and typographical errors. Cassette or disk versions cost just $69.29 with a 20,000 word dictionary. FLEX version $178.58.
75,000 word optional dictionary costs $50 additional.
ALL-IN-ONE editor word processor mailing list program costs $50 (STAR-DOS version) or $75 (FLEX version).

DISK OPERATING SYSTEM

STAR-DOS provides the power of a big DOS with the simplicity of standard R/S disk format. $49.90 for 16K-64K systems.

SYSTEMS SOFTWARE

HUMBUG is the famous 6809 monitor/debugger adapted to the CoCo. $39.95 for 16K or 32K disk or tape systems, $59.95 for 64K systems using STAR-DOS or FLEX. $29.95 for the MC-10.
REMOTERM lets a terminal or modem control the CoCo or MC-10 for $19.95. Disk or cassette.
COMMTERM communications terminal program for the MC-10 costs just $19.95.
NEWTALK reads out memory contents in words through the TV speaker. $20. Disk or cassette.

HOME FINANCE

CHECK 'N TAX lets you check on your bank. Not just a checkbook balancing program, but a help at tax time too. $50, disk only.

EDUCATION

Numerical Methods is a college level course on computer mathematics. $75, disk or cassette.

GAMES

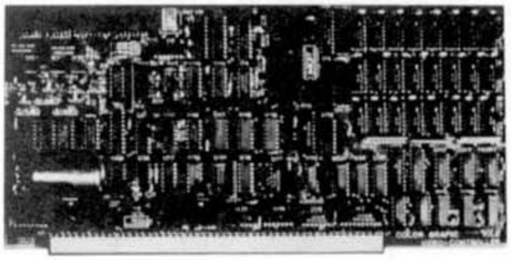
SHRINK is our version of Eliza for $15. Disk or cassette.
THREE-D teaches spatial relationships through three-dimensional tic tac toe. $25, disk or cassette.

STAR-KITS

P.O. BOX 209-M
MT. KISCO, N.Y. 10549
(914) 241-0287

Multi~GRAPH

Multimode-Color-Graphics-Video-Controller



for SS50C-Bus, the ultimate all in one graphics controller:

● 512 512 3 dot high resolution, 14 MHz dot clock ● 192 kbyte on board screen memory for 9 independent pictures ● programmable color-look-up-table for 8 of 4096 progr. colors ● b&w modes include 8 level of greyscale ● X/Y screen-memory-addressing by 12 bit registers, so there are no wrapp-around effects (!) ● sym. X and Y resolution, lines have the same length in X- and Y-direction (!) ● prog. hardw. ASCII char. generator with basic resolution of a 5 8 dots, programmable independent hor. and vert. 1...16 zoom, tilted characters in X and Y direction ● hardw. vector generator with 4 types of lines, 1,500,000 dots/s ● 3 powerfull operation modes, simult.: colored and b/w display, special b&w mode allows selection of any combination of memory-planes to display with no grey scaling (!) ● read-modify-write hardware for non-destructive movement of crosshair(s), cursor(s) or other figures (!) ● fully decoded, jumper set 32 byte addressrange, 16 or 20 bit addressing ● 1 or 2 MHz operation ● indep. programmable hardware-blinking on all 3 planes ● programmable write-protection and display sel. for all planes ● build-in lightpen interface (on dot basis) ● interrupt-generation on IRQ or FIRQ from many sources ● programmable comp. video input allows working with only one monitor for terminal and graphics display (in b&w mode) ● universal RGB and comp.video outputs, CCIR 625 lines, 50 Hz, interl. dependend only on monitor-sync. and not on line frequency (!) nearly all quality RGB monitors are available for this format (!!!) ● designed for easy software implementation, plotter-like programming ● SS50C bus, high quality pc-board, soldermasked and silk screened non standard format 5.5" · 12", but fits into most SWTPC and GIMIX systems (see picture)

Standard delivery is:
high quality pc-board, scematics, manual, EF 9365 data sheet, subroutine package in ASSEMBLER (6809), BASIC, C, PASCAL, FORTRAN77 for easy program interfacing. Source code on FLEX09 compatible floppy.

SS50-Bus board with all features	$ 1670
high speed lightpen with 50 nsec delay and switch	$ 335

special adapters are availible for APPLE and COMMODORE (include case with power supply)
please ask for detailled information

FLEX09 is trademark of SWTPC

Dipl. Ing. Jürgen Knauft
SOFTWARE HARDWARE DIGITALELECTRONIC

D 6457 Maintal 1 (West Germany)
Birkenweg 1 · Tel. 0 61 81 / 4 56 43

GOOD NEWS!

C for the 6809 WAS NEVER BETTER!

INTROL-C/6809,Version1.5

Introl's highly acclaimed 6809 C compilers and cross-compilers are now more powerful than ever!

We've incorporated a totally new 6809 Relocating Assembler, Linker and Loader. Initializer support has been added, leaving only bitfield-type structure members and doubles lacking from a 100% full K&R implementation. The Runtime Library has been expanded and the Library Manager is even more versatile and convenient to use. Best of all, compiled code is just as compact and fast-executing as ever - and even a bit more so! A compatible macro assembler, as well as source for the full Runtime Library, are available as extra-cost options.

Resident Introl-C/6809 compilers running under Flex or OS9 are priced from $375; Uniflex, from $425.

Cross-compilers for PDP-11/Unix hosts are priced from $1500.

Trademarks:
Introl-C, Introl Corporation;
Flex and Uniflex, Technical Systems Consultants;
OS9, Microware Systems;
PDP-11, Digital Equipment Corp.;
Unix, Bell Labs.

For further information, please call or write.

INTROL
CORPORATION
647 W. Virginia St.
Milwaukee, WI 53204
(414) 276-2937

☎ TOLL FREE
1-800-338-6800
For Ordering

SOUTH EAST MEDIA

5900 Cassandra Smith Rd.
Hixson, TN 37343
for information
call (615) 842-4601

CoCo OS-9™ FLEX™
SOFTWARE

ASSEMBLERS

Macro Assembler
the FLEX STANDARD Assembler
F,CCF $50.00

Relocating Assembler w/Linking Loader
Use with many of the C and Pascal Compilers.
F,CCF $150.00

Great Plains Comp. Co.
RMAC
Relocating, Recursive-Macro Assembler
and Linking Loader for the 6809
Supports Recursive Macros, Conditional Assembly, etc. Optional X-Ref Listing. Includes a Small Line-oriented Editor as part of the Assembler.
F,CCF $150.00
F,CCF w/Source $200.00

OmegaSoft
$RALL1
Relocating Assembler and Linking Loader
2-pass Relocating Assembler; 2-pass Linking Loader. Supports 6 Char. Labels, System Calls (SWI[x] FCB x), Expressions with Arith., Logic, and Shifts, etc.
F,CCF $125.00
One Year Maint. $50.00

Windrush Micro Systems
MACE
By Graham Trott. A combined Editor and Assembler designed to allow the Programmer to Enter, Edit, and Assemble Programs with a minimum of effort, w/o leaving the Program. Designed primarily for small to medium sized Program Development. Includes XMACE, a Cross Assembler for the MC6800/1/3 and Hitachi HD6301 (CMOS 6801).
F,CCF - $98.00

DISASSEMBLERS

Computer Systems Consultants
SUPER SLEUTH
Computer Systems Consultants Super Sleuth is a "Time Tested", reliable, PROVEN Disassembler that has gained acceptance through out the SS-50 Bus Community as an extremely POWERFUL, INTERACTIVE, Software Tool. The Super Sleuth Software Package consists of 3 Programs; SLEUTH (the Disassembler), CBGNAM (used to globally Change Labels to a meaningful Name), and XREF (a Cross Reference Generator for Source Code Files). SLEUTH will Disassemble Memory Resident 6809 Code and 6800, 6801, 6802, 6803 (the "Baby CoCo"), 6805, 6808, 6809, and 6502 (Apple, Atari, Commodore, etc.) Binary Disk Files. (See Aug. '83 '68' Micro Journal "Color Users Notes" Column for a full Review.)

Color Computer	**SS-50 Bus (all w/ Source)**
CCD (32K Req'd)	
Obj. Only $49.00	F, $99.00
CCF, Obj. Only $50.00	U, $100.00
CCF, w/Source $99.00	O, $101.00
CCO, Obj. Only $50.00	

Computer Systems Center
DYNAMITE +
An "easy to use", powerful Disassembler for Disk Resident 6809 and 6800 Binary Files. Allows the development of a "Control File" of various Program "Boundaries" during successive disassemblies; can use a Label File which automatically replaces a Hex location with a Label Name; includes an XREF Utility; etc. Label Files provided for Mini-FLEX, FLEX2, FLEX9, Color Computer (for use with Color FLEX Systems), etc. OS-9 Version includes special OS-9 options.

CCF,	Obj.	Only	$100.00
CCO,	"	"	$150.00
F,	"	"	$100.00
O,	"	"	$150.00
U,	"	"	$200.00

COMPILERS AND DECOMPILERS

6809 "Structured" Assembly Lang. Compilers

Windrush Micro Systems
PL/9
By Graham Trott. A "Structured" Assembly Language Editor/Compiler/Debugger, all in ONE PACKAGE; provides a totally INTERACTIVE Program Development Cycle. The Compiler supports Large Symbol Names, Variable Types, Pointers, Control Structures, Stack, A-, B-, and D-Register manipulation, etc. The Source-Oriented Trace/Debugger provides Single Stepping, Break-pointing, etc. An excellent Software Development Tool for utilizing the power of the 6809 in developing small to medium sized packages.
F, CCF - $198.00

PASCAL Compilers

TSC
PASCAL Compiler
Native Code Compiler (UCSD Oriented).
F and CCF - $200.00

Lucidata
PASCAL Compiler
P-Code Compiler (ISO Standard). Designed especially for Microcomputer Systems; Run-time System checks available resources for each task, allowing operation on even minimal computer systems. Allows linkage to Assembler Code for maximum flexibility.
F and CCF 5" - $190.00
F 8" - $205.00

OmegaSoft
PASCAL Compiler
For the PROFESSIONAL; ISO Based, Native Code Compiler. Primarily for Real-Time and Process Control applications. Use custom I/O devices in place of the Pascal INPUT and OUTPUT; Long Int. (32 Bit); Dynamic length strings; Interrupt processing, ROM-able, PIC, Re-Entrant Code, etc. POWERFUL! Includes Source for the Symbolic Debugger, Runtime, and several Utilities. Requires a "Motorola Compatible" Relocating Assembler and Linking Loader.
F and CCF - $425.00
One Year Maint. - $100.00

DECOMPILERS

Southeast Media
DUB (A UniFLEX "basic" De-Compiler)
Re-Create a Source Listing from UniFLEX Compiled basic Programs. Easy to Use; works w/ ALL Versions of UniFLEX basic; Output to Disk or Terminal. Time TESTED and PROVEN; SOLID!
U - $219.95

Introl
C Compiler
A full-featured C, streamlined for the 6809. Generates very efficient object code. Output "benchmarks" close to 10MHz 68000 in 8 Bit Operations; 1.5 times faster than a 4 MHz Z80 when using a 2MHz 6809 System (Re. p 43, "68" Micro Journal, May '83). Floats, etc.
F, CCF, and O - $375.00
U - $425.00
One Year Maint. - $100.00

™FLEX is a trademark of Technical Systems Consultants
™OS9 is a trademark of Microware

☎ TOLL FREE
1-800-338-6800
For Ordering
SOUTH EAST MEDIA
5900 Cassandra Smith Rd.
Hixson, TN 37343
info (615) 842-4601
CoCo OS-9™ FLEX™
SOFTWARE

Availability Legends —
F = FLEX, CCF = Color Computer FLEX
O = OS-9, CCO = Color Computer OS-9
U = UniFLEX
CCD = Color Computer Disk
CCT = Color Computer Tape

Lucidata

COPYCAT

Pascal NOT required

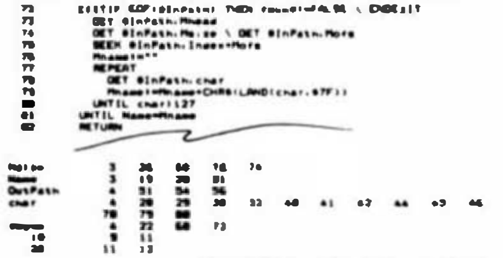
Allows reading TSC Mini-FLEX, SSB DOS68, and Digital Research CP/M Disks while operating under FLEX 1.0, FLEX 2.0, or FLEX 9.0 with 6800 or 6809 Systems. COPYCAT will not perform Miracles, but, between the program and the manual, you stand a good chance of accomplishing a transfer. Includes Utilities to List Directories, Copy Files, and convert Text Files when required. Also includes a Utility for Investigating Physical Compatibility problems. Programs supplied in **Modular Source Code** (Assembly Language) to make it easier to solve unusual problems.

F and CCF 5" - $50.00
F 8" - $65.00

Southeast Media

Basic09 XRef

This **Basic09 Cross Reference Utility** is a Basic09 Program which will produce a "pretty printed" listing with each line numbered, followed by a complete cross referenced listing of all variables, external procedures, and line numbers called. Also included is a **Program List Utility** which outputs the listing without the overhead of building the cross reference table, which allows it to run considerably faster when only a "pretty printed" listing with line numbers is desired. Requires Basic09 or Run8 for operation.



```
72      EXITIF EOF(#InPath) THEN [illegible] \ ENDEXIT
73        GET #InPath,Mhead
74        GET #InPath,Msize \ GET #InPath,More
75        SEEK #InPath,Index+More
76        Mname:=""
77        REPEAT
78          GET #InPath,char
79          Mname:=Mname+CHR$(LAND(char,$7F))
80        UNTIL char>127
81      UNTIL Name=Mname
82      RETURN

Msize        3   26  66  78  76
Name         3   19  30  81
OutPath      4   51  54  56
char         4   28  29  30  33  40  41  43  44  45  46
            78   79  80
[illegible]  4   22  68  73
   10        9   11
   20       11   13
```

O and CCO - Obj. Only - $39.95
O and CCO - w/ Source - $79.95

Southeast Media

O-F

---- OS/9 to FLEX - FLEX to OS/9 ----

Finally: the barrier has been removed between OS/9 and FLEX formatted disks! Now you can READ from and WRITE to, a Single Sided 5" or 8" FLEX diskette from OS-9 with O-F. O-F is a new and unique program, written in BASIC09 (with Source), that performs the following functions:

[illegible]: A BASIC09 Program that reformats a chosen amount of an OS-9 disk to FLEX Format so it can be used normally by FLEX.

FLEX: A BASIC09 Program that does the actual read or write function to the special O-F Transfer Disk, all selected from a user-friendly menu. Functions provided include reading the FLEX Directory, Deleting FLEX Files, Copying both directions, etc. All selections are interactive and complete, including all necessary prompts to the operator.

FLEX users can read, write and use the special disk as any other FLEX disk, provided the FLEX directory is not allowed to continue beyond track zero (too many files).

F and CCF - $79.95

Whimsical Developments

WHIMSICAL

Need the Ease of Design and Maintainability of "Structured Programming" AND the Speed and Control of Assembly Language? Then WHIMSICAL was designed for you! This Single Pass, Recursive Descent Compiler provides the tool for developing simple Utilities to MAJOR Systems in Assembly Language. Supports 3 "Lex" Levels which allow one level of Procedure nesting, or more within "Modules". It is easy to develop programs written for other machines since you are working at the Assembly Language level. Features unified, user-defined I/O; produces ROMable, relocatable, recursive, re-entrant Code; Structured style and statements with Procedures and Modules; supports Byte and Double-Byte primitives with 3 types of Integers (up to 32 bit), Char and Boolean, and unlimited sized Arrays (vectors only); Interrupt handling; unlimited length Variable Names; Variable Initialization (defaults to $00); Include "Source File" directive; Conditional compiling; direct Code insertion; control of the Stack Pointer; etc. To quote Ron Anderson in his review of WHIMSICAL in the Sept. '83 Issue of **'68' Micro Journal** that, except for the lack of floats, "..., **I have to give this one VERY high rating**, ...". It is a FAST Compiler which produces FAST Code (his "Primes" Benchmark ran at **9 secs.** on a 2 MHz System).

F and CCF - $195.00

Southeast Media

COPYMULT

--- Copy LARGE Disks to several smaller disks ---

The following FLEX utilities allow the backup of ANY size disk to any SMALLER size diskettes (Winchester to 8's or 5's, 8" to 5's, etc.). By simply inserting diskettes as requested by COPYMULT, a large disk system may be downloaded to your present floppy disk system, any size. No need to fiddle with directory deletions or any of the other tedious operations that must be done using the normal copy routines.

COPYMULT.CMD understands normal "copy" syntax and always keeps up with files already copied by maintaining directories for both host and receiving disk system, eliminating hours of tedious keyboard entries and other time consuming cleanup chores.

BACKUP.CMD is a special program that downloads "random" type files, any size.

RESTORE.CMD a special program to restructure copied "random" files for copying, or recopying back to the host system.

FREELINK.CMD a "bonus" utility that "relinks" the free chain of floppy or hard disk thereby eliminating fragmentation.

Completely documented source files included.

ALL 4 Programs (8" or 5") $99.50

Southeast Media

CHESS 6809

Requires FLEX and DISPLAYS On Any Type Terminal

Features:

°Four levels of play.
°Swap side. °Point scoring system.
°Two display boards. °Change skill level.
°Solve Checkmate problems in 1-2-3-4 moves.
°Make move and swap sides. °Play white or black.

This is one of the strongest CHESS programs running on any microcomputer, estimated USCF Rating 1600+ (better than most 'club' players at higher levels).

F and CCF - $79.95

™FLEX is a trademark of Technical Systems Consultants
™OS9 is a trademark of Microware

Availability Legends ---

F = FLEX, CCF	= Color Computer FLEX
O = OS-9, CCO	= Color Computer OS-9
U = UniFLEX	
CCD	= Color Computer Disk
CCT	= Color Computer Tape

☎ TOLL FREE
1-800-338-6800
For Ordering

SOUTH EAST MEDIA

5900 Cassandra Smith Rd.
Hixson, TN 37343
for information
call (615) 842-4601

CoCo OS-9™ FLEX™
SOFTWARE

UTILITIES

Southeast Media

DIET-TRAC Forecaster

DIET-TRAC Forecaster is an XBASIC program that plans a diet in terms of either calories and percentage of carbohydrates, proteins and fats (C P G%) or grams of Carbohydrate. Protein and Fat food exchanges of each of the six basic food groups (vegetable, bread, meat, skim milk, fruit and fat) for a specific individual.

Sex, Age, Height, Present Weight, Frame Size, Activity Level and Basal Metabolic Rate for normal individual are taken into account. Ideal weight and sustaining calories for any weight of the above individual are calculated. When a weight goal is given (either gain or loss), and a calorie plan is agreed upon between the computer and the individual, the number of days to reach the weight goal is projected. The starting and ending rate of weight loss is calculated, and a daily calendar with each day's weight for a 30-day period is printed.

F - $59.95
U - $89.95

Southeast Media

XDATA
A COMMUNICATION Package
for the UniFLEX Operating System

Allows UniFLEX Based Systems to Transmit and Receive files to and from other Computer Systems via Modem. Use with CP/M, Main Frames, other UniFLEX Systems, etc.

— Verifies Transmission integrity using checksum or CRC
— Automatically Re-Transmits bad blocks
-- Transmits data in 128 byte blocks

U - $299.99

Southeast Media

JUST
Text Formatter

JUST, a Text Formatter developed by Ron Anderson, provides numerous features which make it a valuable addition to any FLEX Users Software Library. JUST is designed for formatting Text Output for Dot Matrix Printers and provides many unique features:

-Output the "Formatted" Text to the Display for format analysis and change.
-Output the "Formatted" Text to a Text File for use with the supplied PPRINT.CMD for producing multiple copies of the Text on the Printer INCLUDING IMBEDDED PRINTER COMMANDS (this Utility is very useful at other times also, and worth the price of the program by itself).
-"User Configurable" for adapting to other Printers (comes set up for Epson MX-80 with Graftrax); provides for up to ten (10) imbedded "Printer Control Commands", such as Italics on and off, boldface on and off, etc.
-Automatic compensation for a "Double Width" printed line.
-Includes the normal line width, margin, indent, paragraph, space, vertical skip lines, page length, page numbering, centering, fill, justification, etc.
-Use with ANY Editor.
-Supplied with "Structured Source" (Windrush PL/9); easy to see the flow of the program.

F and CCF - $49.95

Lucidata

PASCAL UTILITIES
Requires LUCIDATA Pascal ver 3.

XREF -- produce a Cross Reference Listing of any text; oriented to Pascal Source.

F and CCF - $25.00

INCLUDE -- allows the inclusion of other Files in a Source Text; has unlimited nesting capabilities. Also allows Binary File inclusions.

F and CCF - $25.00

PROFILER -- produces an Indented, Numbered, "Structogram" of a Pascal Source Text File. Allows viewing the overall structure of large programs, and provides clues as to the integrity of the program. Supplied as Source Code; requires compilation.

F and CCF - $25.00

Computer Systems Consultants

FLEX DISK UTILITIES

Eighteen (18) different FLEX Utilities that should be a part of every FLEX Users Toolbox. Ten BASIC Programs to:

Compare, Merge, or Generate Updates between two BASIC Programs, check BASIC Sequence Numbers, compare two unsequenced files, and 5 Programs for establishing a Master Directory of several Disks, and sorting, selecting, updating, and printing paginated listings of these files.

The other 8 Programs provide .CMD Files which allow Copy a File with CRC Errors, so it can possibly be salvaged; Test Disk for errors; Compare two Disks; a fast Disk Backup Program; Edit Disk Sectors; Linearize Free-Chain on the Disk; print Disk Identification; and Sort and Replace the Disk Directory (in sorted order).

ALL Utilities Include Source (either BASIC or Source Code). An EXCELLENT Value!

F and CCF - $50.00

Southeast Media

SPELLB "Computer Dictionary"
OVER 120,000 words!

No more "Let your fingers do the walking through the Dictionary" while you are entering Text with your favorite Editor or Word Processor. SPELLB is more than just "another Spelling Checker"; it allows you to look up a word from within your Editor or Word Processor so that you KNOW it is right WHEN YOU TYPE IT IN with the SPH.CMD Utility (which operates in the FLEX Utility Space). Yes, it ALSO allows you to check and update the Text after you are finished; along with allowing you to ADD WORDS to the Dictionary, "Flag" questionable words in the Text for evaluation later, "View a word in context" before changing or ignoring, etc. SPELLB first checks a "Common Word Dictionary", then the normal Dictionary, then a "Personal Word List", and finally, any "Special Word List" you may have specified. SPELLB also allows the use of Small Disk Storage systems.

F and CCF - $129.95

Great Plains Computer Co.

SPELL

Fast Computer Dictionary — allows directly changing the Text File, adding words to the dictionary, etc. 75,000 words in less than 400 sectors.

F, CCF, OS/9 - $125.00
U - $175.00

VISA
MasterCard

™FLEX is a trademark of Technical Systems Consultants
™OS9 is a trademark of Microware

☎ TOLL FREE
1-800-338-6800
For Ordering
SOUTH EAST MEDIA
5900 Cassandra Smith Rd
Hixson, TN 37343
Info (615) 842-4601
CoCo OS-9™ FLEX™
SOFTWARE

Availability Legends —
F = FLEX, CCF = Color Computer FLEX
O = OS-9, CCO = Color Computer OS-9
U = UniFLEX
CCD = Color Computer Disk
CCT = Color Computer Tape

DATA BASE MANAGEMENT SYSTEMS

Westchester Applied Business Systems
XDMS

Possibly one of the most powerful Database Management Systems' available, this machine language program is small enough to operate on a **single sided 5" disk**, yet provides the **speed** of M.L. and **power** limited only by the user's imagination. This **Relational** DMS supports Sequential, Hierarchical, and Random Access File Structures, and has Virtual Memory capabilities for those Giant Data Bases. Easy-to-use English Language Command Structure.

XDMS — F and CCF - $179.95
XDMS+ (w/ utils) — F and CCF - $249.95
Manual only — $19.95

Westchester Applied Business Systems
XDMS Utilities

These are part of XDMS+, includes SET which provides TTYSET Type of controls within the DMS, PLOT produces horizontal graphs of data, COPYDEF defines new files with existing formats, REVISE allows modification of field definitions, OUTPUT provides an ASCII output of a DMS File, INPUT loads a File from user generated text, PURGE erases all Data from a File, FORM displays the print format, SIZE provides no. of records and sectors info, and FILES outputs a file name matrix of a given file extent.

XDMS Utils. (requires XDMS V1.1) $79.95

Great Plains Computer Co.
INFORAG DBMS

An XBASIC, Menu Driven, DBMS with "Built-In" Audit Tracking, Extremely Powerful Report & Format Capabilities, etc. This **Time Proven** DBMS will become the "Work Horse" of your Software Stable.

F and CCF $295.00
U $395.00

WORD PROCESSORS

Alford and Associates
SCREDITOR III

EXTREMELY Powerful Screen-Oriented Editor/Word Processor. Almost 50 different commands; EXCELLENT Documentation (over 300 pages), including a full Tutorial Section to help you learn how to use the system. Features Cursor-based editing, dynamic Screen Formatting (what you see is what you get), Multi-Column display and editing, "decimal align" columns (AND add them up automatically, if wanted), define multiple keystroke macros, even and odd page number headers and footers, imbed printer control codes in text, full justification series of commands, full "help" support, store common command series on disk for future use, etc. Easy "Set-Up" (for example, you just hit the key you want to use for a specific function, such as "cursor up", and the System reads an stores that key - no digging into tech manuals for codes, etc.); use supplied "set-ups", or remap the keyboard to what you are used too. Except for proportional printing, this package will DO IT ALL!

6800 or 6809 FLEX or SSB DOS, OS-9 - $175.00

Great Plains Computer Co.
STYLOGRAPH

A full-screen oriented WORD PROCESSOR -- (now runs on the Data-Comp and FHL Color FLEX Systems; uses the 51 x 24 Display Screens). Full screen display and editing (i.e., what you see is what you get); supports the Daisy Wheel proportional printers.

SPECIAL CCF - $195.00
F and O - $295.00
U - $395.00

Great Plains Computer Co.
MAIL MERGE

Greatly extends the power and flexibility of STYLOGRAPH. Allows Multiple Text files to be printed out as one large document. Provides for merging information into the Text File during printing (such as different names and addresses), etc.

F, CCF, O - $145.00
U - $195.00

☎ TOLL FREE
1-800-338-6800
For Ordering

SOUTH EAST MEDIA

5900 Cassandra Smith Rd.
Hixson, TN 37343
for information
call (615) 842-4601

CoCo OS-9™ FLEX™
SOFTWARE

ACCOUNTING PACKAGES

Great Plains Computer Co.
Accounting Packages

Accts Rec., Accts Payable & Gen Ledger -- A FULL Accounting Package that can be used together, or as separate packages; provides the IRS required Audit Tracking. (XBASIC, based on the "Osborne Business Programs.")

F and CCF - ea. Program $295.00
U - ea. Program $395.00

Universal Data Research, Inc.
Accounting and Database Manag. Sys.

Note: ALL UDRI Accounting and DBM Progs. Require FLEX and XBASIC. These are Time Tested programs from an old, established, software house.

Color Computer FLEX Systems

DBM Part 1 - $49.95 DBM Part 2 - $49.95
Church Contributions - $49.95
Single Entry Gen Ledger - $49.95
Balanced Billing System - $49.95
A/R $99.95 A/P $99.95 Gen Ledger $189.00
Inventory 2 $69.00 Payroll $99.95

FLEX and UniFLEX — Note: Requires XBASIC or basic

A/P	F - $295,	U - $395
A/R	F - $295,	U - $395
Gen Ledger	F - $295,	U - $395
Inventory 2	F - $295,	U - $395
Payroll	F - $295,	U - $395
DBM	F - $350,	U - $450

Computer Systems Consultants
FULL SCREEN INVENTORY/MRP

The Full Screen Inventory System provides a means of maintaining small inventories. Using a linked, keyed random file structure based upon the item field, it keeps the file in alphabetical order for easier inquiry. With the FIND command, the user may locate and/or print all records matching on partial or complete item, description, vendor, or attributes. Items in backorder or below minimum stock levels may be located and/or printed thru the same process. Printed output may be produced in item or vendor order. A materials requirement planning (MRP) capability for manufacturing environments is included to allow the maintenance and analysis of Hierarchical assemblies of items in the inventory file. It requires TSC's Extended BASIC.

F and CCF - $[illegible], U - $150.00

VISA MasterCard

*FLEX is a trademark of Technical Systems Consultants
*OS9 is a trademark of Microware

☎ TOLL FREE
1-800-338-6800
For Ordering
SOUTH EAST MEDIA
5900 Cassandra Smith Rd.
Hixson, TN 37343
info (615) 842-4601
CoCo OS-9™ FLEX™
SOFTWARE

Availability Legends —
F = FLEX, CCF = Color Computer FLEX
O = OS-9, CCO = Color Computer OS-9
U = UniFLEX
CCD = Color Computer Disk
CCT = Color Computer Tape

☎ TOLL FREE
1-800-338-6800
For Ordering

SOUTH EAST MEDIA

5900 Cassandra Smith Rd.
Hixson, TN 37343
for information
call (615) 842-4601

CoCo OS-9™ FLEX™
SOFTWARE

BUSINESS FORECASTING

The Virginia Company ***Bizpack***

Bizpack is a DESKTOP/PLAN type of Program which is oriented towards **Time Series Analysis** of Business Situations in providing information on which to base common business decisions. This program is used for storing accounting, numeric, and financial data, which can then be used for analyzing, planning, budgeting, forecasting, etc. While "Electronic Spreadsheets" are extremely useful in many situations, Bizpack excels in businesses when there are numerous expense columns, revenue sources, significant business indicators, large numbers, erratic week-to-week and month-to-month fluctuations, etc. Bizpack helps determine statistical relationships, establish trend lines, "smooth" data via moving averages, analyze seasonal data, adjust for inflation, lag data one or more periods in Statistics or Column functions, plot data, etc. The Program displays information on the screen as Columns of information with each Row conforming to a defined Period of Time (weeks, months, years, etc.), and is very easy to use (data is easy to enter, change, and modify; commands can be renamed to suit the users requirements; unlimited ability to create specialized commands using common BASIC Statements; etc.). **Requires TSC's Extended BASIC.**

F and CCF - $135.00
with Source - $250.00

*** SPECIAL ***
Purchase XBASIC and Bizpack together for $221.50
-- a Savings of $13.50 --

COLOR COMPUTER SOFTWARE

Stearns Electronics
FORTH

Intrigued by Forth??? Here is a FORTH package **tailored to the Color Computer!** This package is supplied on Tape, with instructions for transferring it to disk if you wish. Written primarily in machine language, it's **speed** is unparalleled. A full Semigraphic-8 Editor is provided, along with "goodies" like Graphics and Sound Commands, Printer Commands, Auto-Repeat and Control Keys, etc. If you are interested in Learning FORTH, a Trace Feature is provided which is invaluable. If you are a FORTH Pro, this package provides CPU carry Flag accessibility, Fast Task Multiplexing, Clean Interrupt Handling, etc. (Or; you won't "out grow" the Basic capabilities of this Implementation). Combine this package with Leo Brodie's EXCELLENT Book "Starting FORTH", and you will be a FORTH Expert before you know it (and have a lot of fun doing it!).

Color Computer TAPE - $58.95

Custom Software Engineering, Inc.
Color Computer GRAPHIC SCREEN PRINT Programs

Dumps any "PMODE" Screen to the Printer with the BASIC USR Function. Shift the Printout Left or Right or Reverse Print (Dark for Light Screen and Vice Versa). All Programs on Tape.

GSPR	for R.S. LP-VII/VIII & DMP 100/200/400	$7.95
GSPE	for Epson w/ Graftrax and Graftrax +	$9.95
GSPG	for Gemini 10 and 15	$9.95
GSPP	for the Prowriter Printers	$9.95

Custom Software Engineering, Inc.
DATE-O-BASE CALENDAR Program

A Menu Driven EXTENDED BASIC Program which allows the entry of up to 12 Memos per Day, each of which may contain up to 28 Characters, for any day of the Month between the years 1700 and 2099. A Graphic Calendar shows which days contain Memos, and a "Key Word" Search is provided which can be output to the Screen or Printer.

TAPE DATE-O-BASE CALENDAR
(Each Tape File will hold up to 400 Memos) $16.95
DISK DATE-O-BASE CALENDAR
(4,000 Memos at 300/Month per Disk) $19.95

Computer Systems Consultants
TABULA RASA SPREADSHEET

TABULA RASA is similar to DESKTOP/PLAN and provides for the generation and maintenance of tabular computation schemes often used for analysis of business, sales, and economic scenarios. Its menu-driven user interface provides these capabilities even to those users with no programming experience. Its extensive report-generation capabilities allow the user to generate professional results with minimum effort. It requires TSC's Extended BASIC.

F and CCF - $100.00, U - $200.00

Computer Systems Center
DYNACALC

THE Electronic Spread Sheet for 6809 Computer Systems. An extremely POWERFUL Business Tool, this Program will find an unlimited number of "non-business" applications, also (for example, a Full Junior College Electronics Curriculum was set up using DYNACALC). Advanced features like "Table Lookup" make Income Tax work easy; Column or Row Sorting for numerous applications; etc. Completely "Memory Resident", Machine Language, this Program is FAST. Provides STANDARD FLEX Text File output for use with BASIC, Word Processors, Pascal, "C", etc. Also available for Data-Comp and FHL FLEX systems using the 50 x 24 Displays.

F and SPECIAL CCF - $200.00
U - $395.00

ODDS AND ENDS

Computer Systems Consultants
FULL SCREEN FORMS DISPLAY

This package supports any Serial Terminal with cursor control of Memory-Mapped Video Displays. The package substantially extends the screen Input/Output capabilities of TSC's Extended BASIC programs by providing a simple, table-driven method of describing and using full screen displays. These table entries are easy to set up and maintain, and are normally stored on disk and read as required. A simple, interactive means of generating the forms and the data field definitions is provided.

F and CCF - $50.00, U - $75.00

Computer Systems Consultants
FULL SCREEN MAILING LIST

The Full Screen Mailing List System provides a means of maintaining simple mailing lists. Using a random fill structure based on the first character of the name field, it maintains the file in alphabetical order for easier inquiry. With the FIND command, the user may locate all records matching on partial or complete name, city, state, zip, or attributes. Printed listings and output to labels may also be produced on the same selective basis. It requires TSC's Extended BASIC.

F and CCF - $100.00, U - $110.00

VISA MasterCard

™FLEX is a trademark of Technical Systems Consultants
™OS9 is a trademark of Microware

☎ TOLL FREE
1-800-338-6800
For Ordering
SOUTH EAST MEDIA
5900 Cassandra Smith Rd
Hixson, TN 37343
info (615) 842-4601
CoCo OS-9™ FLEX™
SOFTWARE

Availability Legends —
F = FLEX, CCF = Color Computer FLEX
O = OS-9, CCO = Color Computer OS-9
U = UniFLEX
CCD = Color Computer Disk
CCT = Color Computer Tape

Custom Software Engineering, Inc.
That's INTEREST-ing

Interested in INTEREST (the Money Kind)? An EXTENDED BASIC Program that will help you deal with numerous problems requiring interest calculations. Present Value, Rate of Return, Current Bond Yield and Rate of Return to maturity, Loan Repayment Amortization Schedules, etc.

TAPE - $29.95

Custom Software Engineering, Inc.
DISK DATA HANDLER 64K

Custom Software Engineering, Inc.
DISK DOUBLE ENTRY

DISK EXTENDED BASIC Accounting Program w/ Mach. Lang. Routines. A "Traditional" Accounting Package for Small Business, Clubs, Churches, Personal Use, etc. Up to four levels of subtotals with Trial Balance, Income Statement, and Balance Sheet Reports. DDE allows up to 300 Accounts and a Trial Balance of $9,999,999.99. Transactions may be up to 14 lines long, and comments and explanations may be freely used. Accounts are traceable to the journal transaction, which may include comments. Screen reports allow review of past transactions and current Balances.

DISK - $44.95

Computer Systems Center
DYNASHARE
-- Multi-User, Multi-Tasking with FLEX --

Southeast Media is now shipping DYNASHARE FROM STOCK - the multi-user, multi-tasking capability of DYNASHARE allows FLEX users the advantages of more sophisticated and time saving computer usage without having to buy or learn a new Language or Operating System syntax. **DYNASHARE**, as its name implies, allows true "time-sharing" operation under the popular FLEX operating system, and also allows each user to run two simultaneous jobs (multi-tasking); even on single-user systems. For example, while in EDIT, you can list another file or examine a directory. Or, you might look up an item in a Data Base while a Sort is in progress! DYNASHARE also provides some fringe benefits that will be greatly appreciated by FLEX users, including type-ahead, command line editing, and instant response to "escape".

DYNASHARE is the painless method! Use your existing Flex computer by simply adding 64K of RAM for each user and/or task. Fact is, you still use FLEX just like you always have! DYNASHARE is not intended as competition to UniFLEX. It does not improve on the speed of FLEX, and does not offer password protection or other niceties of a full-blown multi-user system. What DYNASHARE does do is give FLEX users **a low-cost way to use existing software in a multi-user, multi-tasking environment**, so your existing FLEX versions of BASIC, XBASIC, editors, assemblers, disassemblers, sort/merge packages, word processors, compilers, DYNACALC spread-sheet package, and so on are still good.

NOTE -- The **initial release** of DYNASHARE is for SWTPC S/09 Computers, but versions will also be available for other popular extended-memory (up to 1024K) systems, such as HELIX and GIMIX. A minimum of 128K of RAM will be required with ALL versions. **DYNASHARE requires 64K of RAM for each active task**; thus a 256K system could allow foreground-background operation on two terminals, or foreground-only operation on four terminals.

AVAILABLE NOW from Southeast Media - $2[illegible]

AUTHORS - PROGRAMMERS
QUALITY SOFTWARE NEEDED
FLEX - UniFLEX - OS/9 - Color Computer

For the past several months, we at the **Southeast Media Division of Computer Publishing, Inc. (CPI)**, the parent company of '68' MICRO JOURNAL and COLOR MICRO JOURNAL, have debated expanding our software distribution business. Many other magazines have been doing so for years (in fact, MOST were in the Software Distribution Business BEFORE they began to publish a Magazine). Presently there are many fine examples of software that has been developed by YOU, our readers, that will never see the "light of day" due to the **Cost of Advertising** and **TIME and Cost involved in the production, distribution, and Customer SUPPORT** of that software unless SOMEONE, with enough exposure and the willingness to continually advertise, runs with the ball.

Software is the "backbone" for the REAL utilization of any Computer System, and ours are no exception! This has been no simple decision. While we realize that there could be some conflict with some of our advertisers, we ALSO hear a LOUD and CONTINOUS cry for **HELP** from our Readers. From day one, the foremost concern of '68' MICRO JOURNAL has been it's **READERS**! Therefore, our **Southeast Media** Division will accept, for appraisal for possible Distribution, 6809 software; Games, Utilities, Software Development, Business Application Programs, etc.

In the past there has been too much software offered that was not quite ready. We will strive to eliminate that element. But, right up front, we tell you only that we will do our very best; nothing more. Also, we will strive to keep cost to a bare minimum, while securing for the author a fair return in royalty payments, promptly paid, and in customer support for his product.

Of course, we will expect, no -- **DEMAND**, that the author keep the product free of errors (bugs), and maintain it in a prompt and business like manner. Also we shall require that authors be willing to furnish 'source' for those programs that justify, by price and utility, inclusion of same. The lack of source code, properly commented, is a continual complaint we hear. Not all programs will be sold with source, but where necessary, we will insist that it be included.

In some instances the program may be small or short and not justify itself as a "single" sale product. In this event it will be combined with other like programs, and offered as a package. In that event, the royalties will be split between the various authors.

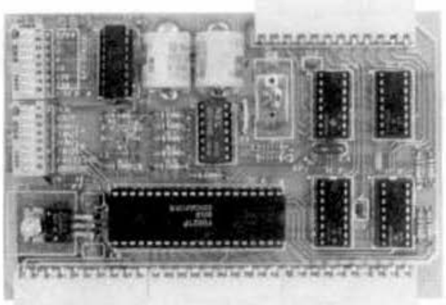
If you have software that you feel will qualify under this program, please contact one of the people below. Remember, if your software has any problems or "funnies" -- **GET IT STRAIGHT BEFORE YOU CONTACT US!!** Also get your source code in proper shape and well commented; there is too much 990 code already drifting around.

If your software is **READY** contact:
Bob Nay, Don Williams, or Tom Williams

Southeast Media is a division
of **Computer Publishing, Inc. (CPI)**,
a family of [illegible] 68XX support facilities.

Availability Legends --
P = FLEX, CCF = Color Computer FLEX
O = OS-9, CCO = Color Computer OS-9
U = UniFLEX
CCD = Color Computer Disk
CCT = Color Computer Tape

CALENDAR-CLOCK / TIMER / PARALLEL PORT



Calendar - Clock CLK68-1

- Keeps date and time whether or not the computer is on
- All clock functions software controlled
- On card battery (included) and charging circuit runs for months
- Day of week, month/day/year, hour/min/sec (12/24 hr)

Interval Timer

- For multi-user operations, multi-tasking, etc.
- Compatible with OS-9** and Flex 2/9*
- Gives time faster with Clock-1 than with timers such as [illegible]
- Generates interrupt intervals from [illegible] to [illegible] sec.

Parallel I/O Port -- Fully buffered 8 bit parallel port

- Dip switches select input or output buffering [illegible] on the board
- Compatible with parallel printer drivers in most versions of BASIC

Construction -- Fully socketed, solder masked, & silk screened

Manual -- Well documented - 36 pages

Dealer & club discount available

Assembled and tested	$119.95	Kit	$89.95
Goldplated bus conn	7.50	2 MHz option	2.50
Disk 5 or 8 in. SSB or Flex* OS.9 Available NOW			14.95

** OS-9 is a trademark of Microware Systems Corporation
* Flex is a trademark of Technical Systems Consultants, Inc.

ROBERTSON ELECTRONICS
1003 Warm Sands Dr. SE
Albuquerque, NM 87123

Phone (505) 294-0025
NM residents add 4% tax
Add $3 Shipping & Handling

COMPARE

our EPROM PROGRAMMER with the field.

All data taken directly from manufacturer's current advertising. Software, interfaces, or personality modules may also be required at additional cost.

- • Triple voltage EPROM
- • Supplied in kit form

		A	B	C	D	E	F
INTERFACE	S30	PAR	PAR	SER	S30	SER	SER
INTELLIGENT	NO	NO	NO	YES	NO	YES	YES
PROGRAMS							
2704•			●				●
2508	●			●	●	●	
2708•			●		●		●
2758	●	●	●	●		●	●
2516	●	●	●	●	●	●	
2716	●	●	●	●	●	●	●
2716•			●		●		●
2532	●		●	●	●	●	●
2732	●		●	●	●	●	●
2732A	●		●		●	●	●
25 4	●		●		●	●	●
2764	●		●		●	●	●
2528	●				●		
27128	●						
2816							●
68764			●				
8748						●	
8749						●	
TOTAL	11	3	12	6	11	11	11
PRICE	$125	$45•	$169	$289	$375	$489	$575

EPRAM EPROM Programmer, $125. Personality module for 2508, 2758, 2516, and 2716 included. Specify CPU, disk size, and operating system (TSC's FLEX or SSB's DOS) when ordering. Manual only, $10: refundable with EPRAM purchase.

UNITEK • P.O. Box 671 • Emporia, VA 23847

SUPER CPU . . . only from LSI

449.95 KIT

We're mighty proud of our new processor card. We're giving you the ability to go 68000 without major changes to your system. Our new CPU gives you these advanced features:

- Dynamic partitioning memory management unit with bound check register.
- On-board timer for multi-user/multi-tasking applications.
- On-board boot-strap EPROM and Monitor EPROM space.
- Vectored priority interrupt generator.
- On-board wait state generator.
- User selectable bus options that includes a new higher bandwidth bus mode.
- And many more . . .

S68K/08-CPU-ASSEM. & TESTED	549.95
S68K/08-CPU-KIT FORM	449.95

(KIT INCLUDES PROCESSOR, CRYSTAL, SOCKETS AND,CONNECTORS)
DISK CONTROLLER SUPPORTED DC3, DC4, DMF2, SDC8.

Announcing . . .

THE *SHELL* FOR FLEX 9™ *

We are pleased to announce the *SHELL*, a UNIX ++ like shell that supports I/O redirection, pipes, macro substitution and programmable shell scripts! The shell will work with all your existing programs and utilities. Requires 56K of user ram, FLEX 9™ * version 2.6 and above. The shell occupies the top 8K of user ram. An excellent tool for the 6809 community.

FLX/SHO9-8	8 inch version	90.00
FLX/SHO9-5	5.25 inch version	90.00
ONE YEAR MAINTENANCE		22.50

LSI STANDS BEHIND ITS PRODUCTS
1 YEAR LIMITED WARRANTY
ON ALL OUR PRODUCTS

ANNOUNCING THE LSI 68000 USERS GROUP

Join the Group by sending us your name and address. You will receive our monthly publication with free public domain user programs and software updates.

New members of the users group will receive a $30.00 discount on THEIR FIRST LSI hardware purchase. Anyone that donates a program to the group will receive our current user group software on a formatted CP/M readable disk.

256K RAM CARD
Using the latest LSI technology this 256K RAM CARD makes a perfect addition to your S68K system. Uses MRDY for refresh arbitration.
S68K/08-256K . . . $750.00

ANY ORDERS RECEIVED ON SDC8 OR THE 256K RAM CARD BEFORE 4-1-84 WILL RECEIVE A $50 DISCOUNT

SDC8 CONTROLLER
A SS50 DMA disk controller for use with either 68000 bus modes or 6809 bus modes. Features a high reliability digital data separator. (No analog circuits to drift) and full 1 Megabyte addressing range.
S68K/08-SDC8 $350.00

CP/M-68K +
DRI's famous CP/M made for the 68000. It includes a C compiler, relocatable assembler, linking loader, librarian and many utilities. It is source compatible with all other CP/M operating systems.
CP/M-68K + . . . $350.00

N.Y. residents add sales tax.

VISA master charge

and COD accepted

*FLEX 9™ is a trademark of Technical Systems Consultants Inc.
+CP/M-68K is a registered trademark of Digital Research, Inc.
++ UNIX is a registered trademark of Bell Labs
All prices and offers subject to change without notice.

LSI Enterprises Ltd.

PO Box 1227
Woodhaven, NY 11421
(212) 423-5596

TEN MOST-ASKED QUESTIONS about DYNACALC™

THE ELECTRONIC SPREAD-SHEET FOR 6809 COMPUTERS

1. What is an electronic spread-sheet, anyway?
Business people use spread-sheets to organize columns and rows of figures. DYNACALC simulates the operation of a spread-sheet without the mess of paper and pencil. Of course, corrections and changes are a snap. Changing any entered value causes the whole spread-sheet to be re-calculated based on the new constants. This means that you can play, 'what if?' to your heart's content.

2. Is DYNACALC just for accountants, then?
Not at all. DYNACALC can be used for just about any type of job. Not only numbers, but alphanumeric messages can be handled. Engineers and other technical users will love DYNACALC's sixteen-digit math and built-in scientific functions. You can build worksheets as large as 256 columns or 256 rows. There's even a built-in sort command, so you can use DYNACALC to manage small data bases — up to 256 records.

3. What will DYNACALC do for ME?
That's a good question. Basically the answer is that DYNACALC will let your computer do just about anything you can imagine. Ask your friends who have VisiCalc™, or a similar program, just how useful an electronic spread-sheet program can be for all types of household, business, engineering, and scientific applications. Typical uses include financial planning and budgeting, sales records, bills of material, depreciation schedules, student grade records, job costing, income tax preparation, checkbook balancing, parts inventories, and payroll. But there is no limit to what YOU can do with DYNACALC.

4. Do I have to learn computer programming?
NO! DYNACALC is designed to be used by non-programmers, but even a Ph.D. in Computer Science can understand it. Even experienced programmers can get jobs done many times faster with DYNACALC, compared to conventional programming. Built-in HELP messages are provided for quick reference to operating instructions.

5. Do I have to modify my system to use DYNACALC?
Nope. DYNACALC uses any standard 6809 configuration, so you don't have to spend money on another CPU board or waste time learning another operating system.

6. Will DYNACALC read my existing data files?
You bet! DYNACALC has a beautifully simple method of reading and writing data files, so you can communicate both ways with other programs on your system, such as the Text Editor, Text Processor, Sort / Merge, STYLOGRAPH™ word processor, RMS™ data base system, or other programs written in BASIC, C, PASCAL, FORTRAN, and so on.

7. How fast is DYNACALC?
Very. Except for a few seldom-used commands, DYNACALC is memory-resident, so there is little disk I/O to slow things down. The whole data array (worksheet) is in memory, so access to any point is instantaneous. DYNACALC is 100% 6809 machine code for blistering speed.

8. Is there a version of DYNACALC for MY system?
Probably. You need a 6809 computer (32k minimum) with FLEX™, UniFLEX™, or OS-9™ operating system. You also need a decent crt terminal, one with at least 80 characters per line, and direct cursor addressing. If your terminal isn't smart enough for DYNACALC, you probably need a new one anyway. The UniFLEX and OS-9 versions of DYNACALC allow you to mix different brands of terminal on the same system. There's also a special version of DYNACALC for Color Computers equipped with FLEX (Frank Hogg or Data-Comp versions).

9. How much does DYNACALC cost?
The FLEX versions are just $200 per copy; UniFLEX version $395; OS-9 version (works with LEVEL ONE or LEVEL TWO) $250. Orders outside North America add $7 per copy for postage. We encourage dealers to handle DYNACALC, since it's a product that sells instantly upon demonstration. Call or write on your company letterhead for more information.

10. Where do I order DYNACALC?
See your local DYNACALC dealer, or order directly from CSC at the address below. We accept telephone orders from 10 am to 6 pm, Monday through Friday. Call us at 314-576-5020. Your VISA or MasterCard is welcome. Please specify diskette size for FLEX or OS-9 versions. Software serial number is required for the UniFLEX version.

Order your DYNACALC today!

Foreign Dealers:

Australia & Southeast Asia: order from Paris Radio Electronics, 161 Bunnerong Road (PO Box 380) Kingsford, 2032 NSW Australia. Telephone: 02-344-9111.

United Kingdom: order from Compusense, Ltd., PO Box 169, London N13 4HT. Telephone: 01-882-0681.

Scandinavia: order from Swedish Electronics hk AB, Murargatan 23-25, Uppsala S-754 37 Sweden. Telephone: 18-25-30-00.

Computer Systems Center
13461 Olive Blvd.
Chesterfield, MO 63017
(314) 576-5020

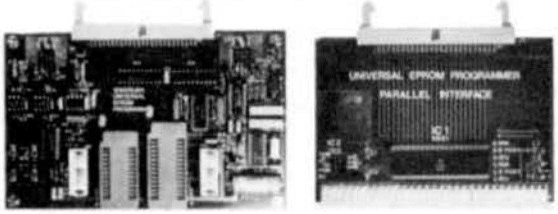
UniFLEX software prices include maintenance for the first year.

DYNACALC is a trademark of
Computer Systems Center

VisiCalc is a trademark of VisiCorp.
STYLOGRAPH is a trademark of Great Plains Computer Co.
RMS is a trademark of Washington Computer Services.
FLEX and UniFLEX are trademarks of TSC.
OS-9 is a trademark of Microware and Motorola.

WINDRUSH MICRO SYSTEMS

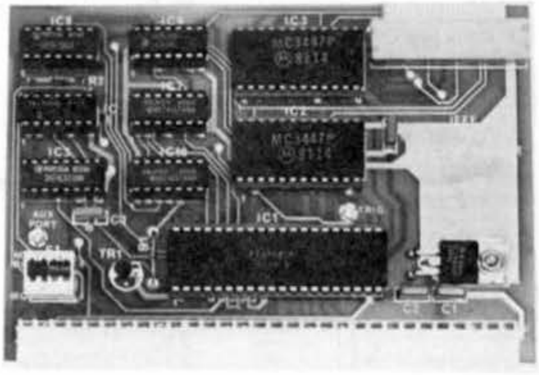
UNIVERSAL EPROM PROGRAMMER



- PROGRAMS and VERIFIES 2508, 2708, 2516, 2716, 2532, 2732A, 2564, and 2764 EPROMS. Minor hardware mods are required to program the INTEL 27128.
- Tri-volt and Single Volt 2508/2708 and 2516/2716 devices are supported.
- ZIF sockets with mode selector switches eliminate 'personality modules'.
- Twin boards with five feet of twisted pair planar cable puts the programmer out on the bench where it belongs.
- SS-30 and EXORCISOR interfaces are available.
- Menu driven software provides the following facilities:
  a. MOVE blocks of memory within the buffer.
  b. READ an EPROM into the buffer.
  c. VERIFY an EPROM against the buffer.
  d. EXAMINE and change the contents of the buffer.
  e. DUMP the contents of the buffer in HEX and ASCII.
  f. FILL a selected area of the buffer with a specified character.
- Software available for all versions of SSB DOS, FLEX 2, FLEX 9 and OS-9. Assembly language source files supplied on disk....enables customising.
- Well documented users manual provides ~~step-by-step~~ adaptation and operating instructions.

AVAILABLE FROM GIMIX IN THE U.S.A.

IEEE-488



- SUPPORTS ALL PRINCIPAL MODES OF THE IEEE-488 (1975/8) BUS SPECIFICATION:

  - Talker - Serial Poll - Single or Dual Primary Address
  - Listener - Parallel Poll - Secondary Address
  - System Controller - Group Trigger - Talk Only...Listen only

- Fully documented with a complete reprint of the KILOBAUD article on the IEEE bus.
- Low level assembly language drivers suitable for 6800, 6801, 6802, 6803, 6808 and 6809 are supplied in the form of listings. These drivers have been extensively tested and are GUARANTEED to work!
- Single SS-30 board (4, 8, or 16 addresses per port), fully socketed, gold plated bus connectors, and IEEE interface cable assembly.

PL/9

EDITOR/COMPILER/DE-BUGGER

- Friendly inter-active environment where you have INSTANT access to the Editor, the Compiler, and the Trace-Debugger, which, amongst other things, can single step the the program a SOURCE line at a time. You also have direct access to any FLEX utility and your System Monitor.
- 250 page manual is organized as a tutorial with plenty of examples.
- Fast single pass compiler produces 8K of COMPACT and FAST 6809 machine code output per minute with no run-time overheads or license fees.
- Fully compatible with TSC text editor format disk files
- Signed and unsigned BYTEs and INTEGERs, 32-bit floating point REALs.
- Vectors (single dimension arrays) and Pointers are supported.
- Mathematical expressions: (+), (-), (*), (/), modulus (\), negation (-)
- Expression evaluators: (=), (<>), (<), (>), (>=), (<=)
- Bit operators: (AND), (OR), (EOR/XOR), (NOT), (SHIFT), (SWAP)
- Logical operators: (.AND), (.OR), (.EOR/.XOR).
- Control statements: IF..THEN..ELSE, IF..CASE1..CASE2..ELSE, BEGIN..END, WHILE.., REPEAT..UNTIL, REPEAT..FOREVER, CALL, JUMP, RETURN, BREAK, GOTO.
- Direct access to (ACCA), (ACCB), (ACCD), (CCR) and (XREG).
- FULLY supports the MC6809 SWI, SWI2, SWI3, NMI, FIRQ, IRQ and RESET vectors. Writing a self-starting (from power-up) program that uses ANY, or ALL, of the MC6809 interrupts is an absolute snap!
- Procedures may be passed and may return variables. This makes them functions which behave as though they were an integral part of PL/9.
- Several fully documented library function modules are supplied: IOSUBS, BITIO, HARDIO, HEXIO, FLEXIO, SCIPACK, STRSUBS, and REALCON.

'... THIS IS THE MOST EFFICIENT COMPILER I HAVE FOUND TO DATE.'

Quoted from Ron Anderson's FLEX User Notes column. Need we say more?

MACE/XMACE

A co-resident EDITOR/ASSEMBLER for the 6809 written by Graham Trott which takes most of the pain out of assembly language program development!

- Friendly inter-active environment where you have INSTANT access to the Editor, the Assembler, FLEX and your System Monitor.
- MACE can also produce ASMPROC's for PL/9 with the assembly language source passed to the output file as comments.
- Includes XMACE a co-resident 6800/1/2/3/8 EDITOR/CROSS ASSEMBLER.

'C'

This is the FLEX version of the James McCosh 'C' compiler that is also available on UNIFLEX from SWTP and OS-9 from Microware!

- The FLEX implementation supports the full Kernighan and Ritchie 'C' specification except 'floats', 'doubles', and 'bit-fields'.
- Produces very efficient assembly language source output with the 'C' source optionally interleaved as comments.
- Built-in optimizer will shorten object code by about 11%
- Supports interleaved assembly language programs.
- The TSC relocating assembler/linking loader (SP09-17) is REQUIRED.

MACE	(6809 FLEX only)..	$ 98.00
PL/9	(6809 FLEX only)...(a steal at this price!)............	$198.00
'C'	(a 56K system and the TSC SP09-17 package is req'd)....	$295.00
IEEE-488	with IEEE-488 cable assembly............................	$298.00
SS-30	Universal EPROM programmer w/one version of software...	$375.00
EXORciser	Universal EPROM programmer w/one version of software...	$395.00
SOFTWARE	Drivers for a 2nd, 3rd, or 4th operating system........	$ 25.00

ALL PRICES INCLUDE AIR MAIL POSTAGE

WE STOCK THE FOLLOWING COMPANIES PRODUCTS: GIMIX, SSB, FHL, MICROWARE, TSC, LUCIDATA, AND ALFORD & ASSOCIATES.

FLEX (tm) is a trademark of Technical Systems Consultants, OS-9 (tm) is a trademark of Microware Systems Corporations, MDOS (tm) and EXORciser (tm) are trademarks of Motorola Incorporated.

An SS-50C all CMOS 256K STATIC RAM board will be available SOON!

Write for details & pricing.

WORSTEAD LABORATORIES,
NORTH WALSHAM, NORFOLK,
ENGLAND. NR28 9SA.
TEL: (0692) 405189
TLX: 97360 SHARET G

FEATURES THE POWERFUL, THIRD GENERATION, MOTOROLA 6809 PROCESSOR!

THE 6809 "UNIBOARD"™

SINGLE BOARD COMPUTER KIT

New!

PERFECT FOR COLLEGES, OEM'S, INDUSTRIAL
AND SCIENTIFIC USES!

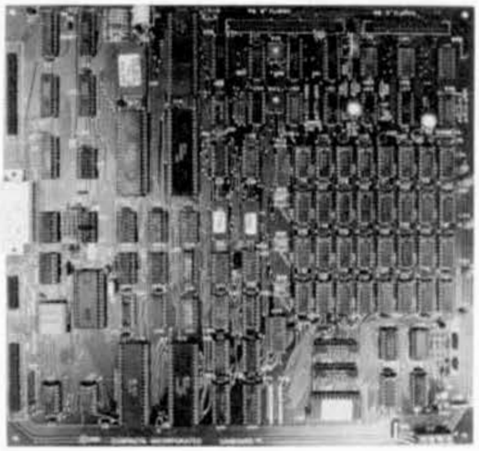
**64K RAM! DOUBLE DENSITY
FLOPPY DISK CONTROLLER!**

BLANK PC BOARD

$99.95

WITH PAL'S, AND
TWO EPROMS.

FOR 5-1/4 OR 8 INCH
SOURCE DISKETTE
ADD $10.



$399.00

COMPLETE KIT!
FULLY SOCKETED.

ALL OPTIONS ARE
STANDARD. NO
EXTRAS TO BUY!

THE COMPACTA UNIBOARD™: Through special arrangement with COMPACTA INC., we are proud to have been selected the exclusive U.S. Mfg. of their new 6809 UNIBOARD™ COMPUTER KIT. Many software professionals feel that the 6809 features probably the most powerful instruction set available today on ANY 8 bit micro. Now, at last, all of that immense computing power is available at a truly unbelievably low price.

ALL SALES ARE MADE SUBJECT TO THE TERMS OF OUR 90 DAY
LIMITED WARRANTY. A FREE COPY IS AVAILABLE UPON REQUEST.

FEATURES:

- ★ 64K RAM using 4116 RAMS.
- ★ 6809E Motorola CPU.
- ★ Double Density Floppy Disk Controller for either 5-1/4 or 8 inch drives. Uses WD1793.
- ★ On board 80 x 24 video for a low cost console. Uses 2716 Char. Gen. Programmable Formats. Uses 6845 CRT Controller.
- ★ ASCII keyboard parallel input interface. (6522)
- ★ Serial I/O (6551) for RS232C or 20 MA loop.
- ★ Centronics compatible parallel printer interface. (6522)
- ★ Buss expansion Interface with DMA channel. (6844)
- ★ Dual timer for real time clock application.
- ★ Powerful on board system monitor (2732). Features commands such as Go To, Alter, Fill, Move, Display, or Test Memory. Also Read and Write Sectors. Boot Normal, Unknown, and General Flex™.

*YOUR CHOICE OF POPULAR
DISK OPERATING SYSTEMS:*

FLEX™ from TSC	$149
OS9™ from Microware	$199

Specify 5-1/4 or 8 Inch

PC BOARD IS
DOUBLE SIDED, PLATED THRU
SOLDER MASKED, 11 x 11-1/2 IN.

Digital Research Computers
(OF TEXAS)
P.O. BOX 461565 • GARLAND, TEXAS 75046 • (214) 271-3536

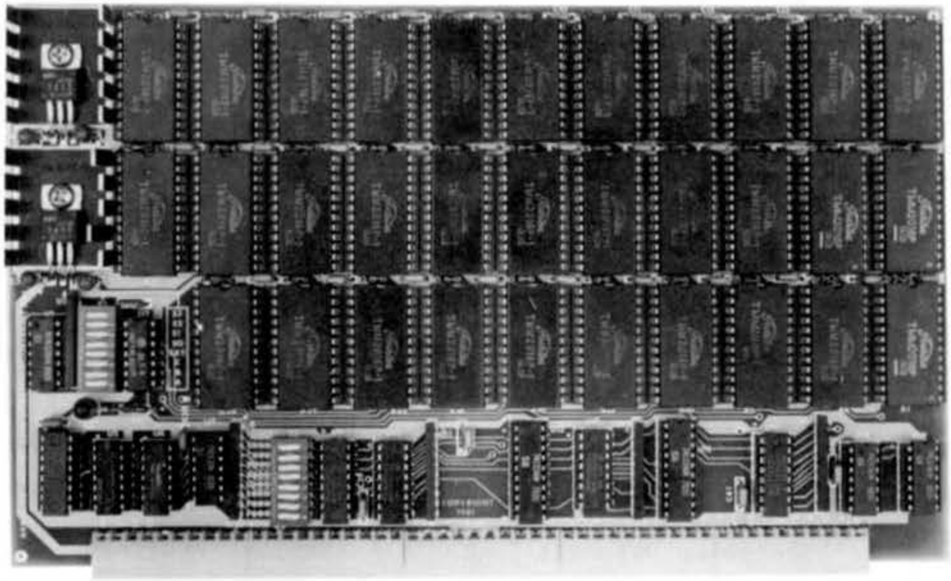
TERMS: Shipments will be made approximately 3 to 6 weeks after we receive your order. VISA, MC, cash accepted. Add $4.00 shipping.
USA AND CANADA ONLY

64K SS-50 STATIC RAM

$179 00 (48K KIT)

NEW!

NEW!



LOW POWER!

RAM OR EPROM!

**BLANK PC BOARD
WITH DOCUMENTATION
$52**

SUPPORT ICs + CAPS -	$18.00
FULL SOCKET SET -	$15.00

ASSEMBLED AND TESTED **ADD $50**

FEATURES:

- ★ Uses new 2K x 8 (TMM 2016 or HM 6116) RAMs.
- ★ Fully supports Extended Addressing.
- ★ 64K draws only approximately 500 MA.
- ★ 200 NS RAMs are standard. (TOSHIBA makes TMM 2016s as fast as 100 NS. FOR YOUR HIGH SPEED APPLICATIONS.)
- ★ Board is configured as 3-16K blocks and 8-2K blocks (within any 64K block) for maximum flexibility.
- ★ 2716 EPROMs may be installed anywhere on Board.
- ★ Top 16K may be disabled in 2K blocks to avoid any I/O conflicts.
- ★ One Board supports both RAM and EPROM.
- ★ RAM supports 2MHZ operation at no extra charge!
- ★ Board may be partially populated in 16K increments.

56K	$219
64K	$249

CLOSE OUT SPECIAL

WE HAVE DROPPED OUR 32K SS-50 STATIC RAM BOARD WHICH USED 2114 LOW POWER RAMS. WE WILL SELL THE REMAINING STOCK OF BLANK PCB'S WITH DATA FOR $17.50 EA. THESE FORMERLY SOLD FOR $50.

16K STATIC RAMS?

The new 2K x 8, 24 PIN. static RAMs are the next generation of high density, high speed, low power. RAMs. Pioneered by such companies as HITACHI and TOSHIBA, and soon to be second sourced by most major U.S. manufacturers, these ultra low power parts, feature 2716 compatible pin out. Thus fully interchangeable ROM/RAM boards are at last a reality, and you get BLINDING speed and LOW power thrown in for virtually nothing.

Digital Research Computers
(OF TEXAS)
P.O. BOX 401565 • GARLAND, TEXAS 75040 • (214) 271-3538

TERMS: Add $2.00 postage. We pay balance. Order under $15 add 75¢ handling. No C.O.D. We accept Visa and MasterCharge. Tex. Res. add 5% Tax. Foreign orders (except Canada) add 20% P & H. Orders over $50, add 85¢ for insurance.

Announcing . . .

THE *SHELL* FOR FLEX 9™

We are pleased to announce the ***SHELL,*** a UNIX™ like shell that supports I/O redirection, pipes, macro substitution and programmable shell scripts! The shell will work with all your existing programs and utilities. Requires 56K of user ram, FLEX 9™ version 2.6 and above. The shell occupies the top 8K of user ram. An excellent tool for the 6809 community.

FLX/SHO9-8. . .8 inch version	90.00
FLX/SHO9-5. . .5.25 inch version . . .	90.00
ONE YEAR MAINTENANCE	22.50

UNIX is a trademark of Bell Labs™.
FLEX 9™ is a trademark of Technical Systems Consultants Inc.

All prices subject to change without notice.

VISA
MC
COD
NY residents add sales tax.

LSI Enterprises Ltd.
PO Box 1227
Woodhaven, NY 11421
(212) 423-5596

MACROPLEX Software

175 Fifth Avenue, Suite 3011, New York, NY 10010

Presenting... **THE CONDUIT**
A new File Type for OS9

THE CONDUIT lets programs exchange data without using external files. Direct program-to-program data transfers on any path use OS9 I/O calls. No need to change device-independent programs. Two-way transfers are permitted and device-dependent code can be accommodated. Programs that formerly ran sequentially can now run concurrently to the limit of memory.

Utilities supplied with **THE CONDUIT** give immediate results when linked to your current programs via **CONDUIT** paths:

- Fast disk storedown speeds up assemblies and other I/O bound procedures
- "In-flight" terminal I/O redirection (e.g., set traps in DEBUG from a disk file)
- Basic09 source inclusions, enhanced listing
- "Null files" drain output, signal EOF on input

Free brochure available. Write, or call (212) 686-3036

Manual alone (50 pages)	$ 8
CONDUIT file system + utilities	$70
Both together	$75
Postage outside North America, add	$ 5

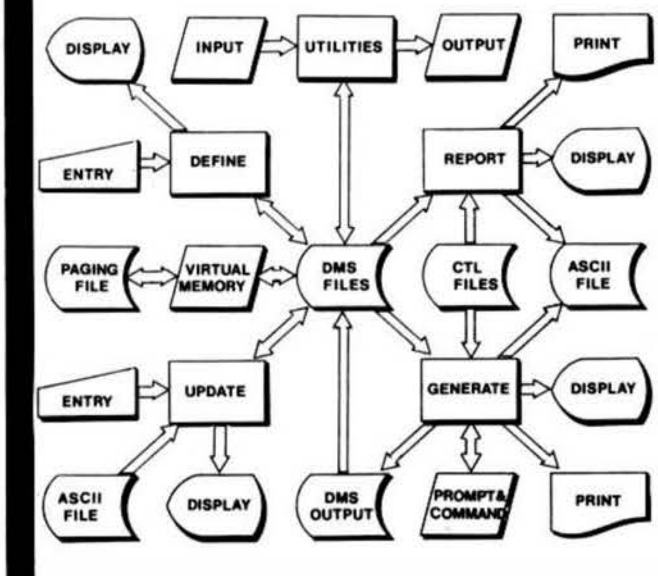
Specify 5" or 8" disk

Version for **OS9 Level Two** available soon

OS9,™ Basic 09,™ Microware Systems Corp. & Motorola, Inc

XDMS

Data Management System



System Architecture

WESTCHESTER Applied Business Systems
Post Office Box 187
Briarcliff Manor, N.Y. 10510

XDMS Data Management System

The XDMS Data Management System is available in three levels. Each level includes the XDMS nucleus, VMGEN utility and System Documentation for level III. XDMS is one of the most powerful systems available for 6800 computers and may be used for a wide variety of applications. XDMS users are registered in our database to permit distribution of product announcements and validation of user upgrades and maintenance requests.

XDMS Level I

XDMS Level I consists of DEFINE, UPDATE and REPORT facilities. This level is intended as an "entry level" system, and permits entry and reporting of data on a "tabular" basis. The REPORT facility supports record and field selection, field merge, sorting, line calculations, column totals and report titling. Control is via a English-like language which is upward compatible with level II. XDMS Level I $129.95

XDMS Level II

Level II adds to Level I the powerful GENERATE facility. This facility can be thought of as a general file processor which can produce reports, forms and form letters as well as file output which may be re-input to the facility. GENERATE may be used in complex processing applications and is controlled by a English-like command language which encompasses that used by Level I. XDMS Level II $199.95

XDMS Level III

Level III includes all of level II plus a set of useful DMS Utilities. These utilities are designed to aid in the development and maintenance of user applications and permit modification of XDMS syste parameters, input and output of XDMS files, display and odification of file format, graphic display of numerical data and other functions. Level III is intended for advanced XDMS users. XDMS Level III $269.95
XDMS System Documentation only ($10. credit toward purchase). . .$ 24.95

XACC Accounting System

The XACC General Accounting System is designed for small business environments of up to 10,000 accounts and inventory items. The system integrates accounting functions and inventory plus the general ledger, accounts receivable and payable functions normally sold separately in other systems. Features user defined accounts, products (or services), transactions, invoicing, etc. Easily configured to most environments.
XACC General Accounting System (Requires XDMS, pref. Lv. III). . $299.95
XACC System Documentation only ($10. credit toward purchase). . .$ 24.95

WESTCHESTER Applied Business Systems
Post Office Box 187, Briarcliff Manor, N.Y. 10510

All software is written in macro/assembler and runs under 6809 FLEX O/S.
Terms: Check, Money Order, Visa or Mastercharge. Shipment first class.
Add P&H $2.50 ($7.50 Foreign). NY Res add sales tax. Specify 5" or 8".

Sales: S. E. MEDIA, 1-800-338-6800, Consultation: 914-941-3552 (evens).

FLEX is a trademark of Technical Systems Consultants, Inc

AMX

Real-Time Multitasking Executive

for
8080, Z80, 6809
and 8086

Gives your application a head start

Save time and money in the development of your product or system by using AMX, the software executive with proven, fault-free operation.

SIMPLE OPERATION

You divide complex control programs into a number of separate, more manageable programs, called *tasks*, each designed to do one job. This allows tasks to be written and tested separately and then combined to form a reliable, finished system.

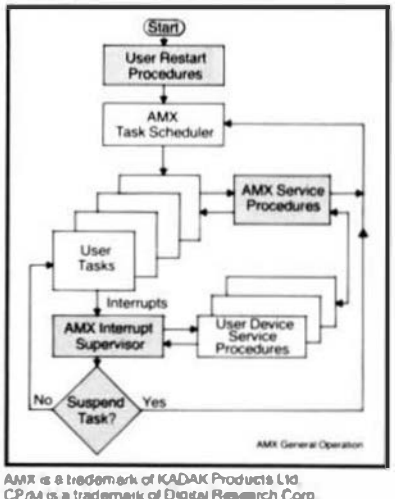
AMX supervises the orderly execution of these tasks, assuring that the most important jobs always get done first. Tasks appear to be executing simultaneously. It's almost like having a separate CPU for each task!

HARDWARE INDEPENDENCE

AMX does not require a particular hardware configuration. *You* control your environment. *You* pick the I/O method. *You* decide the preferred interrupt service technique for your system. AMX will support you on the microprocessor of your choice.

AMX is fast, compact, and ROMable. The AMX nucleus, less than 1400 bytes in size, features multiple task priorities, intertask message passing with priority queuing, external event synchronization, and interval timing.

Support modules provide extended memory management, buffer control and resource allocation. Fast, reentrant integer and floating point math libraries are also available.



AMX General Operation

AMX is a trademark of KADAK Products Ltd
CP/M is a trademark of Digital Research Corp
Z80 is a trademark of Zilog Corp

AMX interfaces support programs written in C, PASCAL, PL/M, FORTRAN and assembler.

Access to CP/M® disk files in real time is possible using the AMX I/O Supervisor.

COMPLETE DOCUMENTATION

We deliver AMX source on diskette to permit AMX to be moved to the software development system of your choice. Our liberal license agreement permits binary (object) distribution without royalties.

HOW TO ORDER

A specification sheet and price list are available, free. Your check or money order for $75 will purchase the AMX Reference Manual for immediate evaluation (specify 8080, Z80, 8086 or 6809 processor). Add $25 for postage and handling outside USA and Canada. The standard 8080/Z80 AMX Multitasking Executive package, including source code, is **$800.** Support modules and interfaces are available separately.

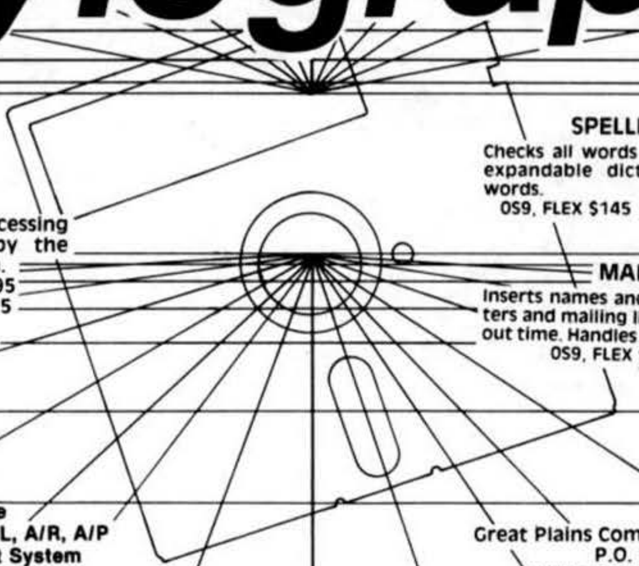
AMX is the choice of professionals the world over. Make it yours, today.

KADAK Products Ltd.

206–1847 W. Broadway Ave., Vancouver, B.C., Canada V6J 1Y5/Phone: (604) 734-2796 Telex: 04-55670

6809 Word Processing System

stylograph™



STYLOGRAPH 2.0

The "User Friendly" word processing system. Fewer key strokes by the operator make it easier to learn.
OS9, FLEX $295 UniFLEX $395
COLOR COMPUTER FLEX $195

SPELLING CHECKER

Checks all words against an internal user-expandable dictionary of over 42,000 words.
OS9, FLEX $145 UniFLEX $195

MAIL MERGE

Inserts names and addresses into form letters and mailing lists. Appends files at print out time. Handles files longer than memory.
OS9, FLEX $125 UniFLEX $175

Inquire about our other software
- **Business Programs - G/L, A/R, A/P**
- **Data Base Management System**
- **Assemblers**

Also, Daisy Wheel Printers $599.

Great Plains Computer Company Inc.
P.O. Box 916
Idaho Falls, Idaho 83401

(208) 529-3210

Flex and Uniflex are trademarks of Technical Systems Consultants, Inc.
OS9 is a trademark of Microware.

AM-100™ ADVANCED WINCHESTER SubSystems for FLEX™ and OS-9™

COMPLETE** AM-100 starting as low as $1899.-

complete** subsystems

5 MB FIXED —	STD. 5" —	$1899.—
10 MB FIXED —	STD. 5" —	$2099.—
5 MB FIXED —	MINI 5" —	$2315.—
5 MB REMOVABLE —	MINI 5" —	$2395.—

** COMPLETE SUBSYSTEM = Drive(s) + Controller + Power Supply + Enclosure + All Cables + Host Interface + Software Drivers NO HIDDEN EXTRAS !!!

options

SS-50 HOST INTERFACE incl. SOFTWARE DRIVERS	$325.—
SET of WINCHESTER SOFTWARE UTILITIES	$49.—

shipping & handling 2.00 (US.Canada) - 4.00 (overseas)

TERMS: • VISA • MONEY ORDER •

Tel.: (514) 737-8787

interfacing technologies corp.

P.O. Box 578, Snowdon, 4890 Bourret Ave. Montreal, Que., H3X 3T7

Trademarks: 1. I.T. Interfacing technologies corp. 2. Technical Systems Consultants Inc. 3. Microware Inc.

QMM2 256K

DYNAMIC MEMORY

A 2Mhz dynamic memory board compatible with SWTPC and other systems, Socketed. assembled, tested, gold connectors, 1 yr. warranty. **$795.00**

OS-9™ SOFTWARE

UTILITIES-A for OS-9 level II includes a fast function PRINTERR, disassembler memory and process dumps, file split and others. **$75.00**

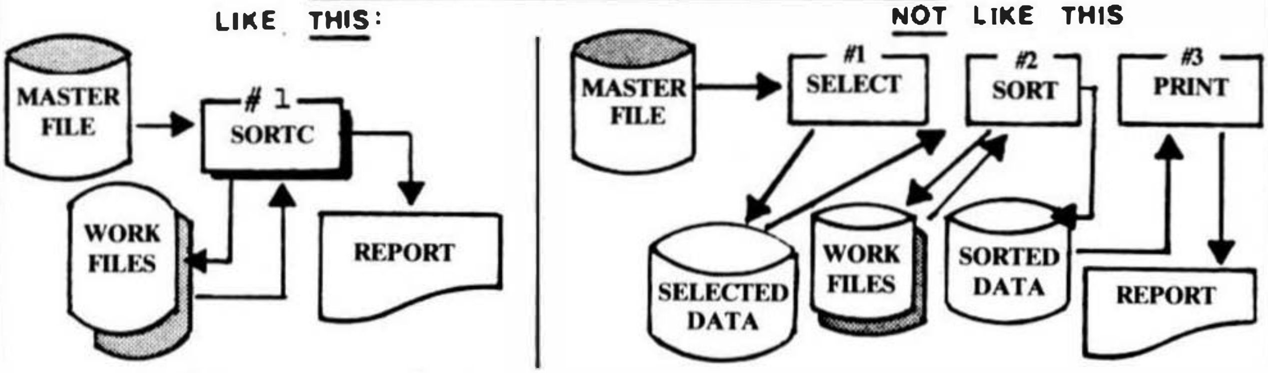
FILTER KIT #1 Contains 11 utilities used as filters to allow wild card directory lists, copies, deletes, moves, lists, pagination, etc. **$40.00**

Terms: Prepaid, COD, VISA, MASTERCARD,
Specify 5¼" or 8" diskette for software.
Send SASE for more info.

D.P. Johnson (503) 244-8152
7655 S.W. Cedarcrest St., Portland, OR 97223
OS-9 is a trademark of Microware and Motorola Inc.

SORTC** for OS9*

THE ONE AND ONLY



BOLDLY GOING WHERE NO SORT HAS GONE BEFORE

SORTC is a high speed, full-record compounding disk sort, which gives microcomputer users mainframe capabilities. It has been specifically designed to sort data efficiently while offering the user great flexibility in designing sort programs. It is written in BASIC09* for use under OS9.

COMPOUNDING FUNCTION

SORTC has the capability of summing user-specified numeric fields on equality of keys. This allows significant savings in memory, disk space, and program development time. A reduction in the number of disk accesses required when compared to other sorts is inherent in the design of SORTC.

DISK BASED

Specifically designed to sort large volumes of data, SORTC imposes no size restrictions on the amount of data to be sorted. It also places no limits on the number of sort keys which can be used or the order in which the keys are sorted. Furthermore, the sort procedure can be performed as many times as necessary within the same program. This feature allows the programmer to take advantage of any existing data bias, and possibly even reduce the size of the sort key.

JBM'S MIDWARE

ADVANCED DESIGN

While most disk sorts are partially based upon the Fibonacci series, SORTC is not. SORTC is a generation ahead of the normal sorts based upon the "Fib series". Its unique algorithm is automatically optimized at run time for a reduction in workspace, reduced # of disk accesses and shorter run times. Designed to be as "crash proof" as possible, the sort procedure will not abort if it is accidentally asked to sort zero items.

EASY TO USE

It is not difficult to design a program which will use JBM's **SORTC**. Since SORTC is a subroutine, the user may write any procedure he or she wants to format the data for sorting and then to process the sorted data. The sorted data need not be written back to disk, but instead is immediately available. The sort code is automatically inserted into the source procedure by a simple Sort Generator.

ORDERING INFORMATION

SORTC, from JBM's MIDWARE line of quality software, is available on either five and one-quarter or eight inch diskettes for a price of $150.00. All of JBM's software packages come complete with comprehensive user's manuals.

For more information, or to place an order, contact:

DEPT. FSEA
The JBM Group, Inc.
332 West Church Road
King of Prussia, PA 19406
TEL: 215-337-3138
TWX: 510-660-3999

the JBM group

VISA and MASTERCHARGE accepted.

*OS9, BASIC09 are registered trademarks of Microware Corporation.

**Uses the same algorithm as JBM's SORTC for Digital Equipment Corp. RSTS Systems.

FREE CATALOG

FRANK
HOGG
LABORATORY

THE REGENCY TOWER • SUITE 215 • 770 JAMES ST. • SYRACUSE, NY 13203
PHONE (315)474-7856 • TELEX 646740

ARCADE 50

POWERFUL COLOR GRAPHI S
Uses the new TMS9918A Video Display procesor. High resolution 256 x 192 pixel display with 15 colors. 16K Bytes of onboard RAM does not reduce user memory. 32 graphic images can be individually moved with simple X-Y commands for smooth animation.
Ext nal Video input allows sublitling.
NTSC composite video output.

SOUND EFFECTS AND MUSIC
- Three AY3-8910 Programmable Sound General rs
- Nine simultaneous voices
- Three independent noise sources
- Onboard stereo amplifier drives two 8 ohm speakers

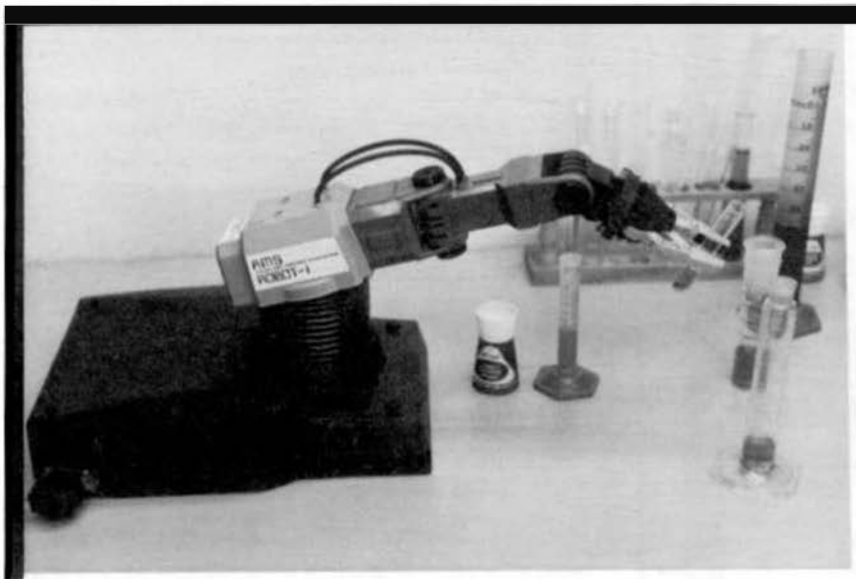
ADDITIONAL I/O CAPABILITIES
- Eight analog inputs with 8 bit resolution
- Supports four joysticks with pushbutton switches
- Eight bit parallel I/O port
- Entire unit maps into 256 bytes of memory

FBASIC

TERMINUS DESIGN INC, in conjunction with Microware Systems Corporation, is proud to announce FBASIC an enhancement of Microware's 6800/BASIC. Their fast compiled BASIC has been adapted for 6809 users with added video and sound features for ARCADE 50 users. FBASIC is a true compiler that produces optimized machine language modules which are ROMable and require no Run-Time package. FBASIC requires less memory overhead and runs hundreds of times faster than BASIC interpreters. It supports standard BASIC instruction including String functions. Disk I/O and fast integer arithmetic with multiple-precision capability Graphics verbs and functions fully support the Arcade 50.

ARCADE 50 assembled and tested	$325.00
Video and Audio connector set	15.00
4 Joystick connector set	15.00
2 Radio Shack Joysticks	24.00
Gold Molex connectors	12.00
A/BASIC for 6800	110.00
FBASIC for 6809	110.00
FBASIC (with ARCADE 50)	75.00
ARCADE 50 RGB	375.00
LABVIDEO (Motorola EXORbus)	375.00
NEW MV09 6809 Processor Board	225.00
256K Dynamic Memory Board	795.00
256K Dynamic Memory Board (w/64K)	395.00
64K D namic Memory Board	295.00

TERMINUS DESIGN INC
16 SCARBROUGH ROAD
ELLENWOOD, GA 30049
[illegible] (404) 474-4866



ROBOT-1

by

ANALOG MICRO SYSTEMS
5660 Valmont Road
Boulder, Co. 80301

Includes SS-30 card with power and 6 channel servo controller, Robot-1 with cables and software.
Order Robot-1s $395. ea.

Interface for 4 joysticks and switches
Order ADS8 $195. ea.

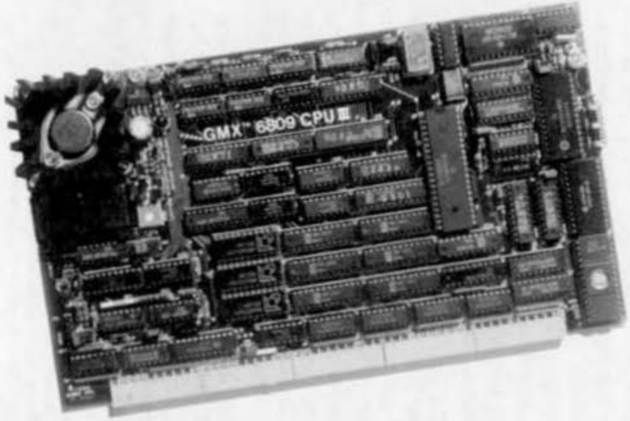
Free Catalog

Up to 8 robot arms with 8 SS-30 servo control cards may be programmed to operate interactively and simultaneously using the AMS multi robot arm software. All movement sequences may be saved on disc and retrieved later for repeating the multi interactive robot arm movement sequences. Source code is provided with all software to enhance the ease of research and experimentation in the field of robotics. Phone (303)444-6809

68' MICRO JOURNAL ADVERTISERS INDEX

This index is provided as a reader service. The publisher does not assume any liability for omissions or errors.

GIMIX STATE OF THE ART 6809 SYSTEMS FOR THE SERIOUS USER.



GIMIX has 19MB or high performance 47MB Winchester Drive Systems and/or Floppy Disk Drive Systems.

For the user who appreciates the need for a bus structured system using STATIC RAM and powered by a ferro resonant constant voltage transformer.

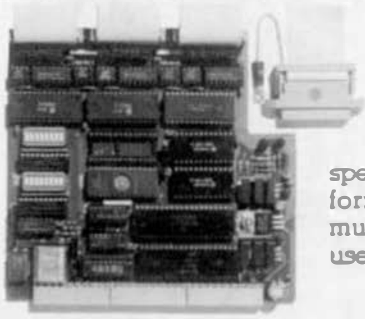
GIMIX has single user systems that can run both FLEX and OS-9 or Multi user systems for use with UniFLEX or OS-9.

GIMIX versions of OS9 and UniFLEX include maintenance and support by Microware (90 days) and TSC (1 year). Maintenance and support after this period are available at extra cost.

(NOTE: this support and maintenance is only for use with approved GIMIX hardware)

For the ultimate in performance, the Unique GMX 6809 CPUIII, using either OS-9-GMXIII or UniFLEX GMXIII (available shortly), gives protection to the system and other users from crashes caused by defective user programs. e.g. During program development, a programmer who crashes goes back to the shell or the debugger, while the other users are not even aware anything occurred.

The intelligent serial I/O processor boards significantly reduce system overhead by handling routine I/O functions, thereby freeing up the host CPU for running user programs. This speeds up system performance and allows multiple terminals to be used at 19.2K baud.



GIMIX 6809 systems support five predominant operating systems:

OS-9 GMX III,
OS-9 GMX II,
UniFLEX,
OS-9 GMX I,
FLEX

and a wide variety of languages and development software.

Whatever your application: software development, instrumentation, process control, educational, scientific or business; whether you need single or multi-user capabilities, GIMIX has hardware and the operating systems to get the job done reliably.

Please phone or write if you need further information.

BASIC-09 and OS-9 are trademarks of Microware Systems Corp. and MOTOROLA, Inc.
FLEX and UniFLEX are trademarks of Technical Systems Consultants, Inc.
GIMIX, GHOST, GMX, CLASSY CHASSIS, are trademarks of GIMIX, Inc.

GIMIX inc.
1337 WEST 37th PLACE • CHICAGO, ILLINOIS 60609 • (312) 927-5510 • TWX 910-221-4055

The Original FLEX™ for Color Computers

- Upgrade to 64K
- RS to FLEX. FLEX to RS file transfer ability
- Create your own character set
- Automatic reognition of single or double density and single or doubled sided
- All features available for either single or multiple drive systems
- Settable Disk Drive Seek Rates
- Faster High Resolution Video Display with 5 different formats
- Save RS Basic from RAM to Disk
- Move RS Basic to RAM
- Load and save function on FLEX disk
- 24 Support Commands 12 with Source Text
- External Terminal Program

DATA-COMP has everything you need to make your TRS-80C Color Computer WORK for YOU; from Parts and Pieces to Full Ready To Use Systems. DATA-COMP designs, sells, services, and SUPPORTS Computer SYSTEMS, not just Software. CALL DATA-COMP TODAY to make your Computer WORK FOR YOU!

System Requirements

FLEX9 Special General Version + Editor & Assembler (which normally sell for $50.00 ea.)	$250.00
F-MATE(RS) FLEX9 Conversion Rout. for the RS Disk Controller	
when purchased with Special General FLEX9 Sys.	$ 49.95
when purchased without the General FLEX9 Sys.	$ 59.95
Color Computer with 64K RAM and EXT.BASIC	$399.95

SPECIAL SYSTEM PACKAGES

64K Radio Shack COLOR COMPUTER, Radio Shack COLOR DISK CONTROLLER, a Disk Drive System, Special General Version of FLEX9, F-MATE(RS), and a Box of 10 Double Density Diskettes, a COMPLETE ready to run SYSTEM on your Color TV Set.

$1099.95

64K Radio Shack COLOR COMPUTER, Radio Shack COLOR DISK CONTROLLER, a Disk Drive System, and a Box of 10 Double Density Diskettes, a COMPLETE ready to run SYSTEM on your Color TV Set.

$799.95

DISK DRIVE PACKAGES, etc.

These Package include the Radio Shack Disk Controller, Disk Drives with Power Supply and Cabinet, and Disk Drive Cable:

PAK #1 - 1 Single Sided, Double Density Sys.	$389.95
PAK #2 - 2 Single Sided, Double Density Sys.	$675.95
PAK #3 - 1 Double Sided, Double Density Sys.	$569.95
PAK #4 - 2 Double Sided, Double Density Sys.	$919.95
PAK #5 - 2 Qume Thinline Double Sided, Double Density Sys.	$699.95
PAK #6 - 2 Tandon Thinline Single Sided, Double Density Sys.	$599.95

PRINTERS

EPSON RX-80	$375.00
EPSON MX-100	$725.00
EPSON MX-80	$395.00
EPSON FX-80	$599.00

SERIAL BOARDS

MX-Series	$119.95
RX-FX-Series	$ 99.95
Spectrum MX-Series	$ 54.95
Mark Data Keyboards	$ 67.95

FLOPPY DISKETTES

	MEMOREX	VERBATIM
5" Soft Sector Disks		
Single Sided Single Density	$2.40ea.	$2.75ea
Single Sided Double Density	$2.40ea	$2.75ea
Double Sided Double Density		$4.92ea
Plastic Storage Box		$2.00ea
8" Soft Sector Disks		
Single Sided Single Density		$3.75ea
Single Sided Double Density		$4.10ea
Double Sided Double Density		$4.75ea
Plastic Library Box		$5.00ea

PARTS AND PIECES

Radio Shack Disk Controller	$169.95
1 Single Sided. Double Density Disk Drive Tandon	$229.95
1 Double Sided. Double Density Disk Drive Qume	$319.95
1 Qume Thinline Double Sided, Double Density	$249.95
1 Tandon 40 Track SSDD with Cabinet and Power Supply	$259.95
Single Drive Cabinet with Power Supply	$ 79.95
Double Drive Cabinet with Power Supply	$109.95
Single Drive Disk Cable for RS Controller	$ 19.95
Double Drive Disk Cable for RS Controller	$ 24.95
Micro Tech. Prods. Inc. LOWER CASE ROM Adapter	$ 74.95
Radio Shack BASIC Version 1.1 ROM or 1.2 ROM	$ 34.95
Radio Shack Extended Basic ROM	$ 74.95
Radio Shack Disk Basic ROM 1.1	$ 29.95
Screen Clean - Clears Up Video Distortion On Your Color Computer	$ 39.95
Set of Eight 64K RAM Chips w Mod Instructions	$ 49.95

For Ordering Call TOLL FREE 1-800-338-6800

LAST OF PRODUCTION!!!

!! LIMITED QUANTITY !!

SWTPC 8212 & 8212(W) Intelligent Terminals

New & Demo Models

At Discount Prices

Remaining Supply of SWTPC 8212 CRT Terminals

EM-82 Video Terminal

Emulates the 8200 Series from SWTPC

DATA-COMP
P.O. Box 794 HIXSON, TN 37343
1-615-842-4601

*FLEX is a trademark of Technical Systems Consultants
**OS9 is a trademark of Microware

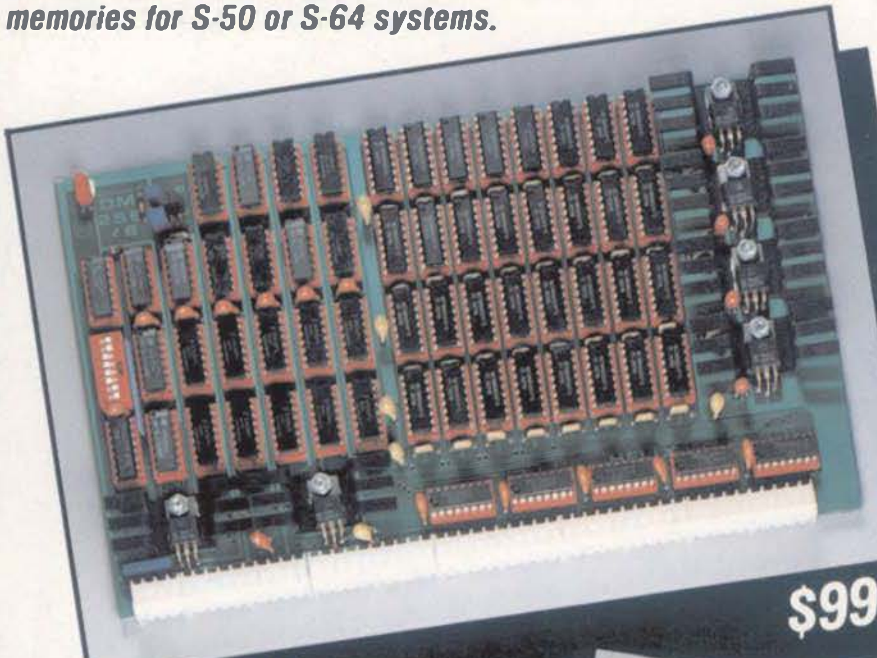
Data-Comp—South East Media & 68 Micro Journal Are Divisions of CPI

68 Micro Journal
5900 Cassandra Smith Rd.
Hixson, TN 37343

Second Class Postage Paid
At Chattanooga, TN
ISSN 0194-5025

MEMORY • MEMORY • MEMORY • MEM

Hazelwood Computer Systems presents a pair of advanced memories for S-50 or S-64 systems.



$995

256K

Dynamic memory with guaranteed 2.5 MHz operation

The DM-256 utilizes the same proven refresh method as our original DM-64. This technique allows fully transparent refresh while maintaining full bus speed and eliminating difficult timing problems.

In addition, individual power regulators for each row of memory ICs enhance reliability by providing the ultimate in noise decoupling. Flexible addressing mode controls make the entire 256K available to OS9 Level 2 or UniFLEX. For FLEX, the furnished utility program HYPERDSK,™ formats memory not used by FLEX as a virtual, superfast disk, thus tremendously improving total system per ormance.

With several hundred already in use, the DM-256 is the best choice for economical, reliable memory.

Assembled, tested, and burned in

Order: DM-256

Static memory with on-board battery backup

Utilizing the latest in low power CMOS static RAMs, the CM-64 has an on-board NiCad battery to maintain memory contents in the absence of bus power. Generous timing margins allow operation at bus speeds in excess of 2 MHz. Zener diode power loss sensing inhibits write operations to prevent data loss during power transitions. Write access can be explicitly controlled by means of an on-board jumper, a remote mounted switch, or an external logic signal.

This allows the memory to appear as ROM to the computer after the contents have been "programmed" using normal computer operations. The economical price makes the advantages of non-volatile memory available to any system.

Assembled, tested, and burned in

Order: CM-64

64K

$395

HAZELWOOD COMPUTER SYSTEMS

907 East Terra, O'Fallon, MO 63366, 314-281-1055

HELIX™

HYPERDSK is a trademark of Hazelwood Computer Systems • OS9 is a trademark of Microware • FLEX and UniFLEX are trademarks of TSC